# OpenStreetMaps Project- Mapping Trenton, Princeton & Plainsboro City Area, New Jersey

### Udacity Data Science Nanodegree course- Final Project

# Data Wrangling with Mongo DB

### By: Mubina Arastu

### MArastu@wgu.edu

# Summary

This project details my analysis and cleanup of the OpenStreetMap.org map data for the majority of Trenton, Princeton & Plainsboro cities of New Jersey. I focused on street addresses and amenities listed within the map.I choose these cities because I live and work in these areas.

**I have used sample OSM data 9.5 MB (9528 kb) and the original size of OSM file (150 MB) for this project. Both the compressed XMLs are provided for the reviewer. This report is based on original XML data.**

This writeup list out few problems that were encountered in the dataset, and gives an high level overview of the data. Exploratory data analysis, wrangling and cleaning of this project can be seen in python notebook.

# Problems With the Dataset

Initialy I faced difficulty in downloading a reasonable size dataset. After few attempts, I finally downloaded sample OSM file that includes Trenton area of New Jersey.

As per my exploratory analysis of the dataset, I found few issues that need to be cleaned up:

1. Over abbreviated city names like (“Twp”,"twp) to clean up.
2. Over abbreviated Street types like ("Blvd") to clean up.
3. Wrong street type like 'Nassau Parl Blvd" to clean up
4. Brand names doesn't have a 'type' tag. "Wikipedia" tag need to be updated as "type".
5. Amenities like shops have some wierd data like 'doityourself" to clean up.

## Check Tag Types & Map Area Boundaries

In [1]:
```
## Create a variable to store xml file name
file='Trenton-Princeton_original.xml'
```

In [2]:
```
import xml.etree.ElementTree as ET
import pprint

#Below code is to find max and minimum latitude and longitude of the map area
def count_tags(filename):
    tags = {}
    bounds = {}
    for event, elem in ET.iterparse(filename):
        if elem.tag == "bounds":
            for latlong in elem.attrib:
                bounds[latlong] = elem.attrib[latlong]
        if elem.tag in tags.keys():
            tags[elem.tag] += 1
        else:
            tags[elem.tag] = 1
    return tags, bounds

## below code is to read from the file
with open(file, 'rb') as mapfile:
    tags, bounds = count_tags(mapfile)
    print ("Found number oftags:")
    pprint.pprint(tags)
    print ("\nMap Area Boundaries:")
    pprint.pprint(bounds)
```

```
Found number oftags:
{'bounds': 1,
 'member': 162351,
 'meta': 1,
 'nd': 831423,
 'node': 686750,
 'note': 1,
 'osm': 1,
 'relation': 1653,
 'tag': 328515,
 'way': 80213}

Map Area Boundaries:
{'maxlat': '40.3887000',
 'maxlon': '-74.4464000',
 'minlat': '40.1860000',
 'minlon': '-74.8543000'}
```

## Check k-values

Below code is to check if there are any k values with special characters and also to see K:Ktype colon seperated pairs. Below code is to check if there are any k values with special characters and also to see K:Ktype colon seperated pairs.

In [7]:
```
import xml.etree.ElementTree as ET
import pprint
import re


## Below code is to check if there are any k values with special characters an
d also to see K:Ktype colon seperated pairs.

lowercase_characters = re.compile(r'^([a-z]|_)*$')
lowercase_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*$')
specialchars = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')
lowercase_colon_vals = {}

##Key_type function search for lower case characters,lower case and colon valu
es and special characters if any.
def key_type(element, keys):
    if element.tag == "tag":
        kval = element.attrib['k']
        if re.search(lowercase_characters, kval):
            keys['lower case'] += 1
        elif re.search(lowercase_colon, kval):
            keys['lower case & colon values'] += 1
            colvals = kval.split(':')
            if colvals[0] not in lowercase_colon_vals.keys():
                lowercase_colon_vals[colvals[0]] = set()
            lowercase_colon_vals[colvals[0]].add(colvals[1])
        elif re.search(specialchars , kval):
            keys['special characters'] += 1
        else:
            keys['other'] += 1

    return keys

## Below function "check_KValues" returns key type element
def check_KValues(filename):
    keys = {"lower case": 0, "lower case & colon values": 0, "special characte
rs": 0, "other": 0}
    for _, element in ET.iterparse(filename):
        keys = key_type(element, keys)

    return keys

## below code is to read from the file
with open(file, 'rb') as mapfile:
    keys = check_KValues(mapfile)
    print ("Types of k-values and their counts:")
    pprint.pprint(keys)
    print ("Types of colon-separated k-values:")
    pprint.pprint(lowercase_colon_vals)
```

```
Types of k-values and their counts:
{'lower case': 204450,
 'lower case & colon values': 114110,
 'other': 9955,
 'special characters': 0}
Types of colon-separated k-values:
{'abandoned': {'highway', 'railway'},
 'access': {'forward', 'delivery'},
 'addr': {'city',
          'country',
          'county',
          'full',
          'housename',
          'housenumber',
          'place',
          'postcode',
          'state',
          'street',
          'suite',
          'unit'},
 'aerodrome': {'type'},
 'aircraft': {'type'},
 'alt_name': {'be', 'vi'},
 'alt_short_name': {'en', 'pl'},
 'area': {'aeroway', 'highway'},
 'artist': {'wikidata'},
 'authentication': {'nfc', 'none', 'app'},
 'beacon': {'code', 'frequency', 'channel', 'type'},
 'brand': {'wikipedia', 'wikidata'},
 'bridge': {'name', 'ref', 'support'},
 'building': {'flats', 'year_built', 'levels', 'part', 'colour', 'use'},
 'capacity': {'disabled'},
 'census': {'population'},
 'charging_station': {'output'},
 'check_date': {'surface'},
 'colour': {'text', 'back', 'arrow'},
 'communication': {'aeronautical_mobile_service',
                   'mast',
                   'mobile_phone',
                   'radio',
                   'satellite',
                   'television'},
 'contact': {'email',
             'facebook',
             'fax',
             'foursquare',
             'instagram',
             'name',
             'phone',
             'twitter',
             'website'},
 'crossing': {'island', 'barrier'},
 'cycleway': {'right', 'both', 'left'},
 'dance': {'teaching'},
 'description': {'en'},
 'destination': {'symbol', 'lanes', 'street', 'ref'},
 'diet': {'vegan', 'vegetarian'},
```

```
'disused': {'amenity', 'shop', 'railway', 'building'},
'drink': {'juice'},
'fire_hydrant': {'type'},
'flag': {'type'},
'fuel': {'biogas', 'lpg', 'cng', 'diesel', 'biodiesel'},
'garden': {'type'},
'generator': {'method', 'source', 'type'},
'gnis': {'county_id',
         'county_name',
         'cr',
         'created',
         'edited',
         'feature_id',
         'feature_type',
         'id',
         'import_uuid',
         'reviewed',
         'state_id'},
'gtfs': {'agency_id'},
'healthcare': {'counselling', 'speciality'},
'heritage': {'operator'},
'hgv': {'conditional', 'national_network', 'state_network', 'minweight'},
'internet_access': {'ssid', 'fee'},
'is_in': {'country',
          'country_code',
          'county',
          'state',
          'state_code',
          'township'},
'isced': {'level'},
'junction': {'ref'},
'kerb': {'height'},
'lanes': {'forward', 'backward', 'both_ways'},
'maxspeed': {'advisory', 'freight', 'type'},
'maxweight': {'signed'},
'memorial': {'type'},
'mtb': {'scale'},
'name': {'ab',
         'ace',
         'af',
         'als',
         'alt',
         'am',
         'an',
         'anc',
         'ang',
         'ar',
         'arc',
         'arz',
         'as',
         'ast',
         'av',
         'ay',
         'az',
         'ba',
         'backward',
         'bar',
```

```
                    'bcl',
                    'be',
                    'bg',
                    'bi',
                    'bm',
                    'bn',
                    'bo',
                    'bpy',
                    'br',
                    'bs',
                    'bxr',
                    'ca',
                    'cdo',
                    'ce',
                    'ceb',
                    'chr',
                    'chy',
                    'ckb',
                    'co',
                    'crh',
                    'cs',
                    'csb',
                    'cu',
                    'cv',
                    'cy',
                    'da',
                    'de',
                    'din',
                    'diq',
                    'dsb',
                    'dv',
                    'dz',
                    'ee',
                    'el',
                    'eml',
                    'en',
                    'eo',
                    'es',
                    'et',
                    'eu',
                    'ext',
                    'fa',
                    'ff',
                    'fi',
                    'fo',
                    'forward',
                    'fr',
                    'frp',
                    'frr',
                    'fur',
                    'fy',
                    'ga',
                    'gag',
                    'gan',
                    'gd',
                    'gl',
                    'glk',
```

```
'gn',
'gu',
'gv',
'ha',
'hak',
'haw',
'he',
'hi',
'hif',
'hr',
'hsb',
'ht',
'hu',
'hy',
'ia',
'id',
'ie',
'ig',
'ik',
'ilo',
'io',
'is',
'it',
'iu',
'ja',
'jbo',
'jv',
'ka',
'kaa',
'kab',
'kbd',
'ki',
'kk',
'kl',
'km',
'kn',
'ko',
'koi',
'krc',
'ks',
'ksh',
'ku',
'kv',
'kw',
'ky',
'la',
'lad',
'lb',
'lbe',
'lez',
'lg',
'li',
'lij',
'lmo',
'ln',
'lo',
'lt',
```

```
                    'ltg',
                    'lv',
                    'lzh',
                    'mdf',
                    'mg',
                    'mhr',
                    'mi',
                    'min',
                    'mk',
                    'ml',
                    'mn',
                    'mo',
                    'mr',
                    'mrj',
                    'ms',
                    'mt',
                    'mwl',
                    'my',
                    'myv',
                    'mzn',
                    'na',
                    'nah',
                    'nan',
                    'nap',
                    'nds',
                    'ne',
                    'new',
                    'nl',
                    'nn',
                    'no',
                    'nov',
                    'nrm',
                    'nso',
                    'nv',
                    'oc',
                    'old',
                    'om',
                    'or',
                    'os',
                    'pa',
                    'pag',
                    'pam',
                    'pap',
                    'pcd',
                    'pdc',
                    'pfl',
                    'pih',
                    'pl',
                    'pms',
                    'pnb',
                    'ps',
                    'pt',
                    'qu',
                    'rm',
                    'rn',
                    'ro',
                    'ru',
```

```
                    'rue',
                    'rw',
                    'sa',
                    'sah',
                    'sc',
                    'scn',
                    'sco',
                    'sd',
                    'se',
                    'sg',
                    'sh',
                    'si',
                    'sk',
                    'sl',
                    'sm',
                    'smn',
                    'sms',
                    'sn',
                    'so',
                    'sq',
                    'sr',
                    'srn',
                    'ss',
                    'stq',
                    'su',
                    'sv',
                    'sw',
                    'szl',
                    'ta',
                    'te',
                    'tet',
                    'tg',
                    'th',
                    'tk',
                    'tl',
                    'tn',
                    'to',
                    'tpi',
                    'tr',
                    'ts',
                    'tt',
                    'tw',
                    'ty',
                    'tzl',
                    'udm',
                    'ug',
                    'uk',
                    'ur',
                    'uz',
                    'vec',
                    'vep',
                    'vi',
                    'vls',
                    'vo',
                    'wa',
                    'war',
                    'wo',
```

```
                'wuu',
                'xal',
                'xh',
                'xmf',
                'yi',
                'yo',
                'yue',
                'za',
                'zea',
                'zh',
                'zu'},
            'network': {'wikipedia', 'wikidata'},
            'nist': {'state_fips', 'fips_code'},
            'notable_tenant': {'wikidata'},
            'note': {'lanes', 'old_railway_operator'},
            'official_name': {'en', 'nl', 'vi', 'cs', 'din', 'eo', 'pl'},
            'old_name': {'en', 'ru', 'vi', 'eo', 'pl'},
            'old_short_name': {'ru'},
            'operator': {'wikidata', 'short', 'wikipedia', 'type'},
            'osmc': {'symbol'},
            'parking': {'fee'},
            'payment': {'american_express',
                        'apple_pay',
                        'bitcoin',
                        'cash',
                        'cheque',
                        'coins',
                        'contactless',
                        'credit_cards',
                        'debit_cards',
                        'discover_card',
                        'ebt',
                        'google_pay',
                        'litecoin',
                        'mastercard',
                        'notes',
                        'snap',
                        'visa',
                        'visa_debit',
                        'wic'},
            'phone': {'alt'},
            'plant': {'method', 'source'},
            'playground': {'theme'},
            'public_transport': {'version'},
            'radio_transponder': {'category', 'callsign'},
            'railway': {'track_ref', 'ref', 'historic', 'traffic_mode', 'position'},
            'ramp': {'wheelchair'},
            'ref': {'fips',
                    'hmdb',
                    'njhrp',
                    'njrhp',
                    'nrhp',
                    'penndot',
                    'retrofitness',
                    'walmart'},
            'restriction': {'conditional', 'hgv'},
            'roof': {'colour', 'levels', 'shape'},
```

```
'short_name': {'en', 'nl', 'pl', 'ru', 'vi', 'zh', 'es', 'cs', 'be', 'mo'},
'social_facility': {'for'},
'socket': {'chademo', 'tesla_supercharger'},
'source': {'alt_name',
           'cycleway',
           'date',
           'geometry',
           'lanes',
           'maxspeed',
           'name',
           'note',
           'old_ref',
           'oneway',
           'population',
           'short_name',
           'surface',
           'unsigned_ref'},
'swing_gate': {'type'},
'theatre': {'type'},
'tiger': {'cfcc',
          'county',
          'name_base',
          'name_direction_prefix',
          'name_direction_suffix',
          'name_type',
          'reviewed',
          'separated',
          'source',
          'tlid',
          'upload_uuid',
          'zip_left',
          'zip_right'},
'toilets': {'wheelchair', 'disposal'},
'tower': {'construction', 'type'},
'traffic_signals': {'direction'},
'turn': {'lanes'},
'was': {'amenity'},
'website': {'state'}}
```

## One of the tag that starts with brand has a subtype as 'wikipedia'. Will change it to 'Type'.

In [8]:
```
import xml.etree.ElementTree as ET
from collections import defaultdict
import pprint

#below function is used to update tag that starts with 'brand' and has a k val
ue of 'wikipedia' to 'type'.
def append_type(filename):
    brands = defaultdict(set)
    for event, elem in ET.iterparse(mapfile, events=("start",)):

        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                k = tag.attrib['k']
                v = tag.attrib['v']
                if k.startswith('brand'):
                    ka = k.split(':')  # create a list of k-values, split on :
(creates array of len 1 if no :)
                    if len(ka) == 1:  # wasn't colon-separated, so was a "typ
e" of lift. Create "type" of lift
                        ka.append('type')
                    if ka[1] == 'wikipedia': # bad tag -- ignore it
                        continue
                    else:
                        brands[ka[1]].add(v)
    return brands

#below code is to read the fie
with open(file, 'rb') as mapfile:
    brands = append_type(mapfile)
    pprint.pprint(brands)
```

```
defaultdict(<class 'set'>,
            {'type': {'7-Eleven',
                      'ALDI',
                      'AMC',
                      'AT&T',
                      'Acme',
                      'Advance Auto Parts',
                      'Alex and Ani',
                      'American Automobile Association',
                      'Ann Taylor',
                      "Applebee's Neighborhood Grill & Bar",
                      'AutoZone',
                      "BJ's Wholesale Club",
                      'BP',
                      'Banana Republic',
                      'Bank of America',
                      'Barnes & Noble',
                      'Baskin-Robbins',
                      'Bed Bath & Beyond',
                      'Benefit',
                      'Best Buy',
                      'Best Western',
                      'Blimpie',
                      "Bob's Discount Furniture",
                      'Bonefish Grill',
                      'Boost Mobile',
                      'Boston Market',
                      'Boys & Girls Club',
                      'Brooks Brothers',
                      'Buffalo Wild Wings',
                      'Burger King',
                      'Buy Buy Baby',
                      'CVS',
                      'CVS Pharmacy',
                      'Chase',
                      'Chevrolet',
                      'Chick-fil-A',
                      "Chili's",
                      'Chipotle',
                      "Chuck E. Cheese's",
                      'Citgo',
                      'Clarion',
                      'Coldwell Banker',
                      'Conoco',
                      'Costco',
                      'Costco Gasoline',
                      'Courtyard',
                      'Cracker Barrel',
                      'DSW',
                      'Dairy Queen',
                      "David's Bridal",
                      'Delta',
                      'Dero',
                      "Dick's Sporting Goods",
                      'Dollar General',
                      'Dollar Tree',
                      "Domino's",
```

```
"Domino's Pizza",
'DoubleTree',
'Dress Barn',
"Dunkin'",
"Dunkin' Donuts",
'Enterprise',
'Extended Stay America',
'Exxon',
'Family Dollar',
'Famous Footwear',
'FedEx Office',
'Five Below',
'Five Guys',
'GMC',
'GameStop',
'Gap',
'Getty',
'Goodwill',
'Goodyear',
'Gulf',
'H&R Block',
'Hampton',
'Hand & Stone Massage and Facial Spa',
'Harbor Freight Tools',
'Hobby Lobby',
'Holiday Inn',
'HomeGoods',
'Homewood Suites',
'Hooters',
'Hyatt Place',
'Hyundai',
'IHOP',
'Investors Bank',
'J.Crew',
'Jackson Hewitt',
'Jiffy Lube',
'KFC',
'KinderCare',
"Kohl's",
'Liberty Tax',
'Life Storage',
'Little Caesars',
'LongHorn Steakhouse',
"Lowe's",
'Lukoil',
'Lululemon',
'MAC Cosmetics',
'Marshalls',
'Mattress Firm',
'Mazda',
"McDonald's",
"Men's Wearhouse",
'MetroPCS',
'Michaels',
'Midas',
"Mimi's Cafe",
"Modell's Sporting Goods",
```

```
'Monro Muffler Brake',
'Nissan',
'Nordstrom Rack',
'Old Navy',
'Olive Garden',
'On The Border',
'Origins',
'Outback Steakhouse',
"P.F. Chang's",
'PNC Bank',
'Panera Bread',
'Paper Source',
'Party City',
'Penske Truck Rental',
'Pep Boys',
'Perkins',
'PetSmart',
'Phillips 66',
'Pier 1 Imports',
'Pioneer',
'Pizza Hut',
'Planned Parenthood',
'Pollo Campero',
'Popeyes',
'Porsche',
'Qdoba',
'Quality Inn',
'QuickChek',
'REI',
'Rainbow',
'Raymour & Flanigan',
'Red Lobster',
'Red Robin',
'Red Roof Inn',
'Regal Cinemas',
'Rent-A-Center',
'Residence Inn',
'Retro Fitness',
"Rita's Italian Ice",
'Rite Aid',
"Romano's Macaroni Grill",
'Ross',
'Santander',
'Shell',
'ShopRite',
'Snipes',
'Sonesta',
'Sonic',
'Sprint',
'Staples',
'Starbucks',
'State Farm',
'Stop & Shop',
'Subway',
'Sunoco',
'Supercuts',
'T-Mobile',
```

```
                'TD Bank',
                "TGI Friday's",
                'TJ Maxx',
                'Taco Bell',
                'Talbots',
                'Target',
                'Tesla Supercharger',
                'The Goddard School',
                'The Home Depot',
                'The UPS Store',
                'Toyota',
                'Tractor Supply Company',
                "Trader Joe's",
                'U-Haul',
                'US Post Office',
                'Ulta Beauty',
                'United States Post Office',
                'Urban Outfitters',
                'Valero',
                'Verizon Wireless',
                'Volkswagen',
                'Walgreens',
                'Walmart',
                'Wawa',
                'Wegmans',
                'Wells Fargo',
                "Wendy's",
                'Westin',
                'Whole Foods Market',
                'YMCA',
                'Zoës Kitchen',
                'eVgo',
                'gmc;buick;pontiac'},
 'wikidata': {'Q1025921',
              'Q1046951',
              'Q1053170',
              'Q1064893',
              'Q1072948',
              'Q1075788',
              'Q10860683',
              'Q11288478',
              'Q1131810',
              'Q1141226',
              'Q1185675',
              'Q1283291',
              'Q1296860',
              'Q1330910',
              'Q1363885',
              'Q1373493',
              'Q1393809',
              'Q1423218',
              'Q145168',
              'Q15109896',
              'Q15109925',
              'Q152057',
              'Q1524184',
              'Q154950',
```

```
'Q157169',
'Q15903261',
'Q1591889',
'Q16198810',
'Q1656230',
'Q17022081',
'Q17085454',
'Q17089620',
'Q177054',
'Q1809448',
'Q191615',
'Q1925685',
'Q1969162',
'Q2007336',
'Q202210',
'Q2078880',
'Q21463374',
'Q214763',
'Q2438391',
'Q244457',
'Q246',
'Q25000269',
'Q2504643',
'Q2553262',
'Q259340',
'Q2624442',
'Q2717882',
'Q2735242',
'Q2895769',
'Q28993',
'Q2923055',
'Q294721',
'Q29570',
'Q298594',
'Q301965',
'Q3045312',
'Q3259007',
'Q327634',
'Q329347',
'Q3307147',
'Q3312613',
'Q3317844',
'Q3375007',
'Q3414933',
'Q341975',
'Q3433273',
'Q3442791',
'Q3552193',
'Q35996',
'Q3658429',
'Q37158',
'Q38076',
'Q38928',
'Q40993',
'Q41171672',
'Q420822',
'Q42306166',
```

```
'Q4492609',
'Q463436',
'Q465751',
'Q4686051',
'Q474379',
'Q4766699',
'Q4781944',
'Q4826087',
'Q483551',
'Q4835754',
'Q487907',
'Q491516',
'Q4926479',
'Q4931582',
'Q4941599',
'Q4943790',
'Q5003352',
'Q509255',
'Q5206207',
'Q5230388',
'Q524629',
'Q524757',
'Q5272601',
'Q5289230',
'Q53268',
'Q533415',
'Q5360181',
'Q5370765',
'Q5421850',
'Q5433101',
'Q5433457',
'Q5455836',
'Q550258',
'Q55613342',
'Q5576260',
'Q5583655',
'Q5627799',
'Q5646230',
'Q5654601',
'Q57405513',
'Q5835668',
'Q584601',
'Q5874938',
'Q5887941',
'Q5890701',
'Q5936320',
'Q603617',
'Q6117132',
'Q61803820',
'Q6192247',
'Q61994955',
'Q620875',
'Q621532',
'Q62274661',
'Q6410551',
'Q65090033',
'Q6541978',
```

```
'Q6643229',
'Q668687',
'Q6702957',
'Q6791878',
'Q6835667',
'Q688825',
'Q6902090',
'Q7091305',
'Q7130852',
'Q7140896',
'Q715583',
'Q7169056',
'Q7191691',
'Q72629292',
'Q7271689',
'Q7284708',
'Q7299290',
'Q7304886',
'Q7304949',
'Q7313497',
'Q7315394',
'Q7336456',
'Q7362714',
'Q738853',
'Q744149',
'Q7501097',
'Q752941',
'Q7561808',
'Q7643239',
'Q7669891',
'Q7679064',
'Q7771029',
'Q78165540',
'Q785943',
'Q7862902',
'Q7880076',
'Q7882701',
'Q795454',
'Q806085',
'Q8074747',
'Q81234570',
'Q813782',
'Q830334',
'Q835638',
'Q839466',
'Q846301',
'Q847743',
'Q864407',
'Q919641',
'Q929722',
'Q942741',
'Q967265'}})
```

# Check Unique User Ids and Names

In [10]:
```
import xml.etree.ElementTree as ET
import pprint
import re

## This function is to get the UID
def get_userid(element):
    return element.attrib['uid']

# This function is to get the User
def get_username(element):
    return element.attrib['user']

# This function is to process map uid and usernames
def get_distinctUsers(filename):
    userids = set()
    usernames = set()
    for _, element in ET.iterparse(filename):
        if 'uid' in element.attrib.keys():
            userid = get_userid(element)
            userids.add(userid)
        if 'user' in element.attrib.keys():
            username = get_username(element)
            usernames.add(username)

    return userids, usernames

with open(file, 'rb') as mapfile:
    userids, usernames = get_distinctUsers(mapfile)
    print ("There are %d unique user IDs" % len(userids))
    print ("There are %d unique usernames" % len(usernames))
```

```
There are 1458 unique user IDs
There are 1458 unique usernames
```

# Auditing Street Types & Cities

## Find if there are any possible street type problems

In [12]:
```
import xml.etree.cElementTree as ET
from collections import defaultdict
import re
import pprint

street_type_re = re.compile(r'\b\S+\.?$', re.IGNORECASE)

## create a list expetced that has expected values for street types
expected = ["Street", "Avenue", "Boulevard", "Drive", "Court", "Place", "Squar
e", "Lane", "Road",
            "Trail", "Parkway", "Commons"]

## below function is used to match street type with 'expected' street type val
ues and find if there are any unexpected vallues.
def audit_street_type(street_types, street_name, street_types_count):
    m = street_type_re.search(street_name)
    if m:
        street_type = m.group()
        if street_type not in street_types_count.keys():
            street_types_count[street_type] = 1
        else:
            street_types_count[street_type] += 1
        if street_type not in expected:
            street_types[street_type].add(street_name)

    return street_types_count

##below function is used to store colon seperatd k value into element
def is_street_name(elem):
    return (elem.attrib['k'] == "addr:street")

## below function returns unexpected street types and unexpected street types
 counts
def audit(mapfile):
    street_types = defaultdict(set)
    street_types_count = {}
    for event, elem in ET.iterparse(mapfile, events=("start",)):

        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                if is_street_name(tag):
                    street_types_count = audit_street_type(street_types, tag.a
ttrib['v'], street_types_count)

    return street_types, street_types_count

##open file
with open(file, 'rb') as mapfile:
    street_types, street_types_count = audit(mapfile)
    print ("Find if there are any street type problems:")
    pprint.pprint(street_types)
    print ("Count of Street Types:")
    pprint.pprint(street_types_count)


##############################################
```

```
 ### below code is to find if any issues with city names
import xml.etree.ElementTree as ET
import pprint

def find_city_errors(filename):
    city = defaultdict(set)
    for event, elem in ET.iterparse(mapfile, events=("start",)):

        if elem.tag == "node" or elem.tag == "way":
            for tag in elem.iter("tag"):
                k = tag.attrib['k']
                v = tag.attrib['v']
                if k.startswith('addr:city'):
                    ka = k.split(':')  # create a list of k-values, split on :
(creates array of len 1 if no :)
                    if len(ka) == 1:
                        if ka[1] == 'city':
                            continue
                    else:
                        city[ka[1]].add(v)
    return brands

##Open file
with open(file, 'rb') as mapfile:
    city = find_city_errors(mapfile)
    pprint.pprint(city)
```

```
Find if there are any street type problems:
defaultdict(<class 'set'>,
            {'1': {'US Highway 1', 'Route 1'},
             '111': {'South Clinton Avenue Ste. 111'},
             '130': {'Route 130',
                     'U.S. 130',
                     'US 130',
                     'US Highway 130',
                     'US Hwy 130',
                     'US Route 130',
                     'Us Highway 130'},
             '160': {'Carnegie Center West, Suite 160'},
             '19067': {'East Trenton Avenue Morrisville, PA 19067'},
             '27': {'Route 27'},
             '29': {'Route 29'},
             '31': {'NJ 31', 'Rt 31'},
             '33': {'Highway 33',
                    'NJ 33',
                    'New Jersey State Highway 33',
                    'Route 33',
                    'Rte 33',
                    'State Highway 33'},
             '683': {'County Road 683'},
             'Ave': {'1655 North Olden Ave',
                     'Alden Ave',
                     'N Olden Ave',
                     'Parkway Ave',
                     'Pennington Ave',
                     'S Clinton Ave',
                     'S. Clinton Ave',
                     'Sloan Ave',
                     'West Trenton Ave',
                     'Yard Ave'},
             'Ave.': {'Broad Ave.'},
             'B': {'Princeton Ave #B'},
             'Blvd': {'Floral Vale Blvd',
                      'Nassau Park Blvd',
                      'Nassau Parl Blvd',
                      'Route 1 South and Promenade Blvd',
                      'Summerfield Blvd'},
             'Center': {'Overlook Center'},
             'Cir': {'Blossom Cir'},
             'Circle': {'Berwick Circle',
                        'Compton Circle',
                        'Erica Circle',
                        'Moorsgate Circle'},
             'Cres': {'Ely Cres'},
             'Crossing': {"Washington's Crossing"},
             'Ct': {'Rousillon Ct', 'Heller Park Ct'},
             'Dr': {'Clubhouse Dr',
                    'Lawrence Dr',
                    'Merrill Lynch Dr',
                    'Old Stone Mill Dr',
                    'Wood Mill Dr',
                    'Wyncrest Dr'},
             'E': {'College Rd E'},
             'East': {'460 Route 33 East',
```

```
          'College Road East',
          'Palmer Square East'},
 'Extension': {'Spruce Street Extension'},
 'H': {'Robbinsville Allentown Rd Ste H'},
 'Harrison': {'Harrison'},
 'Highway': {'Lincoln Highway'},
 'Ln': {'Olden Ln'},
 'N': {'US 130 N', 'US-1 N', 'Route 31 N'},
 'NJ-31': {'NJ-31'},
 'NJ-33': {'NJ-33'},
 'North': {'Greenfield Drive North'},
 'PIke': {'Princeton PIke'},
 'Pike': {'Princeton Pike', 'Brunswick Pike'},
 'Plaza': {'Riverview Plaza'},
 'Rd': {'Campus Rd',
        'Cranbury Rd',
        'Ridge Rd',
        'Robbinsville Allentown Rd',
        'Robbinsville-Allentown Rd',
        'Schalks Crossing Rd',
        'Whitehorse Hamilton Square Rd'},
 'Rd.': {'Lawrence Rd.'},
 'Robbinsvil': {'Allentown Robbinsvil'},
 'S': {'US-1 S'},
 'South': {'US 1 South', 'Forrestal Road South'},
 'St': {'Alexander St',
        'Cooper St',
        'E State St',
        'Market St',
        'N Broad St',
        'N Main St',
        'Nassau St',
        'S Broad St'},
 'Terrace': {'Queensboro Terrace', 'Lakeview Terrace'},
 'US-1': {'US-1'},
 'US-130': {'US-130'},
 'W': {'International Road W'},
 'Way': {'Allerton Way',
         'Azalea Way',
         'Barto Way',
         'Beethoven Way',
         'Bristol Way',
         'Brookfield Way',
         'Capital Way',
         'Carriage Way',
         'Chelten Way',
         'Durham Way',
         'Fulham Way',
         'Hickory Way',
         'Independence Way',
         'John Fitch Way',
         'Justice Samuel A Alito Jr Way',
         'New Canton Way',
         'Nottingham Way',
         'Petal Way',
         'Rembrandt Way',
         'Roosevelt Way',
```

```
                          'Sculptors Way',
                          'Southern Way',
                          'Spencer Way',
                          'Strathmore Way',
                          'Sturges Way',
                          'Sullivan Way',
                          'Sylvia Way',
                          'Walt Whitman Way',
                          'West Manor Way',
                          'Western Way',
                          'Windward Way'},
                   'West': {'College Road West', 'Carnegie Center West'},
                   'al': {'pullen al'},
                   'ave': {'Stuart ave',
                           'and 1891 brunswick ave',
                           'hillcrest ave',
                           'michigan ave'},
                   'avenue': {'Ohio avenue'},
                   'dr': {'hampshire dr'},
                   'ext': {'brunswick ave ext'},
                   'rd': {'Lawrenceville rd'},
                   'st': {'centre st',
                          'clay st',
                          'e front st',
                          'e state st',
                          'jackson st',
                          'livingston st',
                          'market st',
                          'mercer st',
                          'pear st',
                          's broad st',
                          's montgomery st',
                          's stockton st',
                          's warren st',
                          'st',
                          'w state st'}})
Count of Street Types:
{'1': 3,
 '111': 1,
 '130': 9,
 '160': 1,
 '19067': 1,
 '27': 3,
 '29': 1,
 '31': 2,
 '33': 19,
 '683': 3,
 'Ave': 12,
 'Ave.': 1,
 'Avenue': 2020,
 'B': 1,
 'Blvd': 5,
 'Boulevard': 153,
 'Center': 1,
 'Cir': 1,
 'Circle': 88,
 'Court': 417,
```

```
 'Cres': 2,
 'Crossing': 1,
 'Ct': 2,
 'Dr': 37,
 'Drive': 887,
 'E': 1,
 'East': 12,
 'Extension': 44,
 'H': 1,
 'Harrison': 1,
 'Highway': 3,
 'Lane': 451,
 'Ln': 1,
 'N': 3,
 'NJ-31': 1,
 'NJ-33': 3,
 'North': 1,
 'PIke': 2,
 'Pike': 343,
 'Place': 94,
 'Plaza': 4,
 'Rd': 10,
 'Rd.': 1,
 'Road': 1525,
 'Robbinsvil': 1,
 'S': 2,
 'South': 2,
 'Square': 1,
 'St': 11,
 'Street': 1772,
 'Terrace': 25,
 'US-1': 1,
 'US-130': 2,
 'W': 2,
 'Way': 308,
 'West': 2,
 'al': 3,
 'ave': 25,
 'avenue': 192,
 'dr': 1,
 'ext': 1,
 'rd': 11,
 'st': 458}
defaultdict(<class 'set'>,
            {'type': {'7-Eleven',
                      'ALDI',
                      'AMC',
                      'AT&T',
                      'Acme',
                      'Advance Auto Parts',
                      'Alex and Ani',
                      'American Automobile Association',
                      'Ann Taylor',
                      "Applebee's Neighborhood Grill & Bar",
                      'AutoZone',
                      "BJ's Wholesale Club",
                      'BP',
```

```
'Banana Republic',
'Bank of America',
'Barnes & Noble',
'Baskin-Robbins',
'Bed Bath & Beyond',
'Benefit',
'Best Buy',
'Best Western',
'Blimpie',
"Bob's Discount Furniture",
'Bonefish Grill',
'Boost Mobile',
'Boston Market',
'Boys & Girls Club',
'Brooks Brothers',
'Buffalo Wild Wings',
'Burger King',
'Buy Buy Baby',
'CVS',
'CVS Pharmacy',
'Chase',
'Chevrolet',
'Chick-fil-A',
"Chili's",
'Chipotle',
"Chuck E. Cheese's",
'Citgo',
'Clarion',
'Coldwell Banker',
'Conoco',
'Costco',
'Costco Gasoline',
'Courtyard',
'Cracker Barrel',
'DSW',
'Dairy Queen',
"David's Bridal",
'Delta',
'Dero',
"Dick's Sporting Goods",
'Dollar General',
'Dollar Tree',
"Domino's",
"Domino's Pizza",
'DoubleTree',
'Dress Barn',
"Dunkin'",
"Dunkin' Donuts",
'Enterprise',
'Extended Stay America',
'Exxon',
'Family Dollar',
'Famous Footwear',
'FedEx Office',
'Five Below',
'Five Guys',
'GMC',
```

```
                    'GameStop',
                    'Gap',
                    'Getty',
                    'Goodwill',
                    'Goodyear',
                    'Gulf',
                    'H&R Block',
                    'Hampton',
                    'Hand & Stone Massage and Facial Spa',
                    'Harbor Freight Tools',
                    'Hobby Lobby',
                    'Holiday Inn',
                    'HomeGoods',
                    'Homewood Suites',
                    'Hooters',
                    'Hyatt Place',
                    'Hyundai',
                    'IHOP',
                    'Investors Bank',
                    'J.Crew',
                    'Jackson Hewitt',
                    'Jiffy Lube',
                    'KFC',
                    'KinderCare',
                    "Kohl's",
                    'Liberty Tax',
                    'Life Storage',
                    'Little Caesars',
                    'LongHorn Steakhouse',
                    "Lowe's",
                    'Lukoil',
                    'Lululemon',
                    'MAC Cosmetics',
                    'Marshalls',
                    'Mattress Firm',
                    'Mazda',
                    "McDonald's",
                    "Men's Wearhouse",
                    'MetroPCS',
                    'Michaels',
                    'Midas',
                    "Mimi's Cafe",
                    "Modell's Sporting Goods",
                    'Monro Muffler Brake',
                    'Nissan',
                    'Nordstrom Rack',
                    'Old Navy',
                    'Olive Garden',
                    'On The Border',
                    'Origins',
                    'Outback Steakhouse',
                    "P.F. Chang's",
                    'PNC Bank',
                    'Panera Bread',
                    'Paper Source',
                    'Party City',
                    'Penske Truck Rental',
```

```
'Pep Boys',
'Perkins',
'PetSmart',
'Phillips 66',
'Pier 1 Imports',
'Pioneer',
'Pizza Hut',
'Planned Parenthood',
'Pollo Campero',
'Popeyes',
'Porsche',
'Qdoba',
'Quality Inn',
'QuickChek',
'REI',
'Rainbow',
'Raymour & Flanigan',
'Red Lobster',
'Red Robin',
'Red Roof Inn',
'Regal Cinemas',
'Rent-A-Center',
'Residence Inn',
'Retro Fitness',
"Rita's Italian Ice",
'Rite Aid',
"Romano's Macaroni Grill",
'Ross',
'Santander',
'Shell',
'ShopRite',
'Snipes',
'Sonesta',
'Sonic',
'Sprint',
'Staples',
'Starbucks',
'State Farm',
'Stop & Shop',
'Subway',
'Sunoco',
'Supercuts',
'T-Mobile',
'TD Bank',
"TGI Friday's",
'TJ Maxx',
'Taco Bell',
'Talbots',
'Target',
'Tesla Supercharger',
'The Goddard School',
'The Home Depot',
'The UPS Store',
'Toyota',
'Tractor Supply Company',
"Trader Joe's",
'U-Haul',
```

```
                    'US Post Office',
                    'Ulta Beauty',
                    'United States Post Office',
                    'Urban Outfitters',
                    'Valero',
                    'Verizon Wireless',
                    'Volkswagen',
                    'Walgreens',
                    'Walmart',
                    'Wawa',
                    'Wegmans',
                    'Wells Fargo',
                    "Wendy's",
                    'Westin',
                    'Whole Foods Market',
                    'YMCA',
                    'Zoës Kitchen',
                    'eVgo',
                    'gmc;buick;pontiac'},
 'wikidata': {'Q1025921',
              'Q1046951',
              'Q1053170',
              'Q1064893',
              'Q1072948',
              'Q1075788',
              'Q10860683',
              'Q11288478',
              'Q1131810',
              'Q1141226',
              'Q1185675',
              'Q1283291',
              'Q1296860',
              'Q1330910',
              'Q1363885',
              'Q1373493',
              'Q1393809',
              'Q1423218',
              'Q145168',
              'Q15109896',
              'Q15109925',
              'Q152057',
              'Q1524184',
              'Q154950',
              'Q157169',
              'Q15903261',
              'Q1591889',
              'Q16198810',
              'Q1656230',
              'Q17022081',
              'Q17085454',
              'Q17089620',
              'Q177054',
              'Q1809448',
              'Q191615',
              'Q1925685',
              'Q1969162',
              'Q2007336',
```

```
'Q202210',
'Q2078880',
'Q21463374',
'Q214763',
'Q2438391',
'Q244457',
'Q246',
'Q25000269',
'Q2504643',
'Q2553262',
'Q259340',
'Q2624442',
'Q2717882',
'Q2735242',
'Q2895769',
'Q28993',
'Q2923055',
'Q294721',
'Q29570',
'Q298594',
'Q301965',
'Q3045312',
'Q3259007',
'Q327634',
'Q329347',
'Q3307147',
'Q3312613',
'Q3317844',
'Q3375007',
'Q3414933',
'Q341975',
'Q3433273',
'Q3442791',
'Q3552193',
'Q35996',
'Q3658429',
'Q37158',
'Q38076',
'Q38928',
'Q40993',
'Q41171672',
'Q420822',
'Q42306166',
'Q4492609',
'Q463436',
'Q465751',
'Q4686051',
'Q474379',
'Q4766699',
'Q4781944',
'Q4826087',
'Q483551',
'Q4835754',
'Q487907',
'Q491516',
'Q4926479',
'Q4931582',
```

```
'Q4941599',
'Q4943790',
'Q5003352',
'Q509255',
'Q5206207',
'Q5230388',
'Q524629',
'Q524757',
'Q5272601',
'Q5289230',
'Q53268',
'Q533415',
'Q5360181',
'Q5370765',
'Q5421850',
'Q5433101',
'Q5433457',
'Q5455836',
'Q550258',
'Q55613342',
'Q5576260',
'Q5583655',
'Q5627799',
'Q5646230',
'Q5654601',
'Q57405513',
'Q5835668',
'Q584601',
'Q5874938',
'Q5887941',
'Q5890701',
'Q5936320',
'Q603617',
'Q6117132',
'Q61803820',
'Q6192247',
'Q61994955',
'Q620875',
'Q621532',
'Q62274661',
'Q6410551',
'Q65090033',
'Q6541978',
'Q6643229',
'Q668687',
'Q6702957',
'Q6791878',
'Q6835667',
'Q688825',
'Q6902090',
'Q7091305',
'Q7130852',
'Q7140896',
'Q715583',
'Q7169056',
'Q7191691',
'Q72629292',
```

```
'Q7271689',
'Q7284708',
'Q7299290',
'Q7304886',
'Q7304949',
'Q7313497',
'Q7315394',
'Q7336456',
'Q7362714',
'Q738853',
'Q744149',
'Q7501097',
'Q752941',
'Q7561808',
'Q7643239',
'Q7669891',
'Q7679064',
'Q7771029',
'Q78165540',
'Q785943',
'Q7862902',
'Q7880076',
'Q7882701',
'Q795454',
'Q806085',
'Q8074747',
'Q81234570',
'Q813782',
'Q830334',
'Q835638',
'Q839466',
'Q846301',
'Q847743',
'Q864407',
'Q919641',
'Q929722',
'Q942741',
'Q967265'}})
```

## Following issues are found from above output: I will handle these issues in Mongo DB.

1. Abbreviated terms like: twp,Twp. These should be changed to Township.
2. Wrong spelling found: Nassau Parl Blvd- It should be Nassau Park Boulevard.

## First clean up data and then save into JSON file format.¶

In [14]:
```
import xml.etree.ElementTree as ET
import pprint
import re
import codecs
import json


lower = re.compile(r'^([a-z]|_)*$')
lower_colon = re.compile(r'^([a-z]|_)*:([a-z]|_)*$')
problemchars = re.compile(r'[=\+/&<>;\'"\?%#$@\,\. \t\r\n]')

CREATED = [ "version", "changeset", "timestamp", "user", "uid"]


def shape_element(element):
    node = {}
    pos = ['lat', 'lon']
    created = ['version', 'changeset', 'timestamp', 'user', 'uid']
    if element.tag == "node" or element.tag == "way" :

        node['created'] = {}
        node['type'] = element.tag
        for attrib, value in element.attrib.items():
            if attrib in pos:
                if 'pos' not in node.keys():
                    node['pos'] = [0,0]  # only create position sub-document i
f position is one of the attributes
                if attrib == 'lat':
                    node['pos'][0] = float(value)
                else:
                    node['pos'][1] = float(value)
            elif attrib in created:
                node['created'][attrib] = value
            else:
                node[attrib] = value

# Format addresses into subdocument
        for tag in element.iter("tag"):
            k = tag.attrib['k']
            v = tag.attrib['v']
            if k.startswith('addr'): # Format addresses into subdocument
                addrtag = k.split(':')
                if len(addrtag) > 2:
                    continue
                if 'address' not in node.keys():
                    node['address'] = {}
                node['address'][addrtag[1]] = v
#Below code is to Format brand into types
            elif k.startswith('brand'):
                atag = k.split(':')  # create a list of k-values, split on :
                if len(atag) == 1:
                    atag.append('type')
                if atag[1] == 'wikipedia': # bad tag -- ignore it
                    continue
                if 'brand' not in node.keys():
                    node['brand'] = {}
```

```
                node['brand'][atag[1]] = v
                #else:
                #node[k] = v

        for tag in element.iter("nd"):
            if 'node_refs' not in node.keys():
                node['node_refs'] = []
            ref = tag.attrib['ref']
            node['node_refs'].append(ref)

    return node

## Write to Json file- write corrected xml into JSON
def create_JSON(file_in, pretty = False):
    file_out = "Trenton_Princeton_original_Maps.json".format(file_in)
    data = []
    with codecs.open(file_out, "w") as fo:
        for _, element in ET.iterparse(file_in):
            el = shape_element(element)
            if el:
                data.append(el)
                if pretty:
                    fo.write(json.dumps(el, indent=2)+"\n")
                else:
                    fo.write(json.dumps(el) + "\n")
    return data



with open(file, 'rb') as mapfile:
    data_array = create_JSON(mapfile, False)
```

## Auditing ends here. Data is loaded into JSON. Rest of the data cleanup will be done in Mongo DB.

# MongoDB

## Open Connection to MongoDB

In [15]:
```
#import modules
import pymongo
from pymongo import MongoClient
import pprint

#Open Connection
client = MongoClient('mongodb://127.0.0.1:27017/?directConnection=true&serverS
electionTimeoutMS=2000')
```

# Open Database

In [16]:
```
#Open Database
db = client.db_Trenton_Princeton
db = client['db_Trenton_Princeton']
print(db)
```

```
Database(MongoClient(host=['127.0.0.1:27017'], document_class=dict, tz_aware=
False, connect=True, directconnection=True, serverselectiontimeoutms=2000),
'db_Trenton_Princeton')
```

# Print Collections

In [17]:
```
#Print Collections
collections = db.list_collection_names()
print ("collections:", collections, "\n")
```

```
collections: ['TrentPrincemap', 'TrentPrincemaps']
```

### For this project we will be using collection: TrentPrincemaps

### Lets clean up street types and city errors found while auditing

In [128]:
```
# Using update_one function to update 'twp' to Township
city=db.TrentPrincemaps
city.update_many({'address.city': 'Hopewell twp'}, {'$set': {'address.city':
'Hopewell Township'}})
city.update_many({'address.city': 'Ewing twp'}, {'$set': {'address.city': 'Ewi
ng Township'}})
city.update_many({'address.city': 'Princeton twp'}, {'$set': {'address.city':
'Princeton Township'}})
city.update_many({'address.city': 'West windsor twp'}, {'$set': {'address.cit
y': 'West Windsor Township'}})
city.update_many({'address.city': 'Lawrence Twp'}, {'$set': {'address.city':
'Lawrence Township'}})
```

Out[128]:
```
<pymongo.results.UpdateResult at 0x2b804570300>
```

In [126]:
```
#Get all cities
cityreplaced  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$address.city",
         "count" : {"$sum" : 1}}},
          { '$sort' : { 'count' : -1 } }
                                ])
for k in cityreplaced:
    print(k)
```

```
{'_id': None, 'count': 810098}
{'_id': 'Trenton city', 'count': 1587}
{'_id': 'Lawrence Township', 'count': 1503}
{'_id': 'Trenton', 'count': 1379}
{'_id': 'Monroe', 'count': 876}
{'_id': 'East Windsor', 'count': 865}
{'_id': 'Robbinsville Township', 'count': 376}
{'_id': 'Hopewell Township', 'count': 238}
{'_id': 'Ewing Township', 'count': 164}
{'_id': 'Princeton', 'count': 107}
{'_id': 'Robbinsville', 'count': 101}
{'_id': 'Princeton Township', 'count': 93}
{'_id': 'East Windsor Township', 'count': 89}
{'_id': 'Hamilton Township', 'count': 81}
{'_id': 'Plainsboro', 'count': 76}
{'_id': 'Hightstown', 'count': 68}
{'_id': 'Hamilton', 'count': 42}
{'_id': 'Hamilton Square', 'count': 38}
{'_id': 'Lawrenceville', 'count': 36}
{'_id': 'Dayton', 'count': 33}
{'_id': 'Pennington', 'count': 33}
{'_id': 'Ewing', 'count': 27}
{'_id': 'Yardley', 'count': 25}
{'_id': 'West Windsor Township', 'count': 22}
{'_id': 'West Windsor', 'count': 15}
{'_id': 'Cranbury Township', 'count': 11}
{'_id': 'Monmouth Junction', 'count': 11}
{'_id': 'Hopewell', 'count': 8}
{'_id': 'Jamesburg', 'count': 6}
{'_id': 'Monroe Township', 'count': 6}
{'_id': 'Princeton Junction', 'count': 6}
{'_id': 'South Brunswick', 'count': 6}
{'_id': 'Cranbury', 'count': 6}
{'_id': 'Morrisville', 'count': 5}
{'_id': 'Mercerville', 'count': 3}
{'_id': 'Kingston', 'count': 3}
{'_id': 'Robbinsville Twp', 'count': 3}
{'_id': 'Fairless Hills', 'count': 3}
{'_id': 'Windsor', 'count': 3}
{'_id': 'princeton', 'count': 2}
{'_id': 'West Trenton', 'count': 2}
{'_id': 'ROBBINSVILLE', 'count': 1}
{'_id': 'Plainsboro Township', 'count': 1}
{'_id': 'Robbisnville', 'count': 1}
{'_id': 'Levittown', 'count': 1}
```

In [136]:
```
# Using update_many function to clean street error
street=db.TrentPrincemaps
street.update_many({'address.street': 'Nassau Parl Blvd'}, {'$set': {'address.
street': 'Nassau Park Boulevard'}})
street.update_many({'address.street': 'Nassau Park Blvd'}, {'$set': {'address.
street': 'Nassau Park Boulevard'}})
```

Out[136]:
```
<pymongo.results.UpdateResult at 0x2b8045f0380>
```

### Find Documents

In [127]:
```
db.TrentPrincemaps.count_documents({})
```

Out[127]:
```
818060
```

## Find Nodes

In [19]:
```
#Find Nodes
nodes=db.TrentPrincemaps.count_documents({'type':'node'})
pprint.pprint("count of nodes")
pprint.pprint(nodes)
```

```
'count of nodes'
732944
```

## Find Ways

In [20]:
```
#Find Ways
ways=db.TrentPrincemaps.count_documents({'type':'way'})
pprint.pprint("count of ways")
pprint.pprint(ways)
```

```
'count of ways'
85116
```

## Find unique Users

In [9]:
```
u=len(db.TrentPrincemaps.distinct('created.user'))
pprint.pprint("Number of Unique Users")
pprint.pprint(u)
```

```
'Number of Unique Users'
167
```

# Top 5 Users

In [21]:
```
topusers  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$created.user",
         "count" : {"$sum" : 1}}},
          { '$sort' : { 'count' : -1 } }, { '$limit' : 5 }
                              ])
for k in topusers:
    print(k)
```

```
{'_id': 'RM23', 'count': 91641}
{'_id': 'woodpeck_fixbot', 'count': 70444}
{'_id': 'NJDataUploads', 'count': 46779}
{'_id': 'OceanVortex', 'count': 31600}
{'_id': 'Utible', 'count': 29535}
```

# Find Amenities & Highways

In [22]:
```
#Find Service Highways
s=db.TrentPrincemaps.count_documents({"highway": "service"})
pprint.pprint("count of services")
pprint.pprint(s)

#Find Residential Highways
r=db.TrentPrincemaps.count_documents({"highway": "residential"})
pprint.pprint("count of residential areas")
pprint.pprint(r)

#Find number of parking sites
p=db.TrentPrincemaps.count_documents({"amenity": "parking"})
pprint.pprint("count of parking areas")
pprint.pprint(p)
```

```
'count of services'
1656
'count of residential areas'
237
'count of parking areas'
216
```

# List number of shops

In [26]:
```
## Aggregate function is used to show olny top 30 shops based on counts sorted
in descending order
shops  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$shop",
         "count" : {"$sum" : 1}}},
          { '$sort' : { 'count' : -1 } }, { '$limit' : 30 }
                              ])
for sh in shops:
    print(sh)
```

```
{'_id': None, 'count': 817927}
{'_id': 'furniture', 'count': 15}
{'_id': 'car', 'count': 12}
{'_id': 'clothes', 'count': 12}
{'_id': 'department_store', 'count': 12}
{'_id': 'variety_store', 'count': 6}
{'_id': 'mobile_phone', 'count': 6}
{'_id': 'shoes', 'count': 6}
{'_id': 'craft', 'count': 6}
{'_id': 'supermarket', 'count': 6}
{'_id': 'alcohol', 'count': 6}
{'_id': 'car_repair', 'count': 6}
{'_id': 'cosmetics', 'count': 3}
{'_id': 'houseware', 'count': 3}
{'_id': 'tyres', 'count': 3}
{'_id': 'baby_goods', 'count': 3}
{'_id': 'party', 'count': 3}
{'_id': 'music', 'count': 3}
{'_id': 'outdoor', 'count': 3}
{'_id': 'sports', 'count': 3}
{'_id': 'massage', 'count': 3}
{'_id': 'pet', 'count': 3}
{'_id': 'hairdresser', 'count': 3}
{'_id': 'electronics', 'count': 3}
{'_id': 'copyshop', 'count': 3}
{'_id': 'doityourself', 'count': 1}
```

## From above data one wierd name 'doityourself' is seen as a shop category. Will change doityourself to 'home_improvement'.

In [27]:
```
# Using update_one function to update 'doityourself' to home_improvement
coll=db.TrentPrincemaps
updateResult = coll.update_one({'shop': 'doityourself'}, {'$set': {'shop': 'ho
me_improvement'}})
```

## check if 'doityourself is updated to home_improvement with 1 count'

In [28]:
```
## Using aggregate function with limit to show only 10 shops which is sorted f
rom least to highest

updatedshops  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$shop",
         "count" : {"$sum" : 1}}},
          { '$sort' : { 'count' : 1 } }, { '$limit' : 10}
                                ])
for upsh in updatedshops:
    print(upsh)
```

```
{'_id': 'home_improvement', 'count': 1}
{'_id': 'massage', 'count': 3}
{'_id': 'cosmetics', 'count': 3}
{'_id': 'music', 'count': 3}
{'_id': 'houseware', 'count': 3}
{'_id': 'outdoor', 'count': 3}
{'_id': 'party', 'count': 3}
{'_id': 'pet', 'count': 3}
{'_id': 'sports', 'count': 3}
{'_id': 'tyres', 'count': 3}
```

In [29]:
```
#Count total number of amenities grouped by each amenity and display in  desce
nding sorting order.
#*************************
print('Total Number of Amenities from highest to lowest counts')
from bson.son import SON
s=db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$amenity",
         "count" : {"$sum" : 1}
         }},  { '$sort' : { 'count' : -1 } }, { '$limit' : 20 }

])
for k in s:
    print(k)
```

```
Total Number of Amenities from highest to lowest counts
{'_id': None, 'count': 817748}
{'_id': 'parking', 'count': 216}
{'_id': 'restaurant', 'count': 39}
{'_id': 'charging_station', 'count': 9}
{'_id': 'fast_food', 'count': 6}
{'_id': 'bank', 'count': 6}
{'_id': 'bicycle_parking', 'count': 6}
{'_id': 'bench', 'count': 6}
{'_id': 'waste_basket', 'count': 6}
{'_id': 'car_rental', 'count': 6}
{'_id': 'cafe', 'count': 3}
{'_id': 'atm', 'count': 3}
{'_id': 'shelter', 'count': 3}
{'_id': 'place_of_worship', 'count': 3}
```

# Additional Exploration

## Number of Offices

In [30]:
```
off=len(db.TrentPrincemaps.distinct('office'))
pprint.pprint("Number of Unique offices")
pprint.pprint(off)
```

```
'Number of Unique offices'
3
```

## List out Offices

In [31]:
```
topoffices  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$office",
         "count" : {"$count" : {}}}},

                                ])
for k in topoffices:
    print(k)
```

```
{'_id': 'company', 'count': 6}
{'_id': 'energy_supplier', 'count': 3}
{'_id': 'government', 'count': 9}
{'_id': None, 'count': 818042}
```

## List out Users Posts

In [32]:
```
topusers  = db.TrentPrincemaps.aggregate([
    {"$group" :
        {"_id" : "$created.user",
         "count" : {"$count" : {}}}},
     { '$sort' : { 'count' : -1 } }
            ])
for k in topusers:
    print(k)
```

```
{'_id': 'RM23', 'count': 91641}
{'_id': 'woodpeck_fixbot', 'count': 70444}
{'_id': 'NJDataUploads', 'count': 46779}
{'_id': 'OceanVortex', 'count': 31600}
{'_id': 'Utible', 'count': 29535}
{'_id': 'ogm17', 'count': 29339}
{'_id': 'James Michael DuPont', 'count': 26672}
{'_id': 'iskra32', 'count': 25389}
{'_id': 'Aurimas Fišeras', 'count': 22894}
{'_id': 'AdamP511', 'count': 15459}
{'_id': 'arielatom', 'count': 12743}
{'_id': 'mblumber', 'count': 11366}
{'_id': 'pezizomycotina', 'count': 11074}
{'_id': 'pipigong', 'count': 10944}
{'_id': 'N8Rtz', 'count': 9231}
{'_id': 'wireguy', 'count': 9231}
{'_id': 'Nikhil K', 'count': 8677}
{'_id': 'multifoil', 'count': 8310}
{'_id': 'abergs95', 'count': 7815}
{'_id': 'NE2', 'count': 7712}
{'_id': 'rjb4095', 'count': 7599}
{'_id': '-Matthew', 'count': 7252}
{'_id': 'mario824', 'count': 6834}
{'_id': 'Ben Sudano', 'count': 5448}
{'_id': 'Chad Bachman', 'count': 5324}
{'_id': 'jpjohnsonjr', 'count': 5162}
{'_id': 'Matt1993', 'count': 5143}
{'_id': 'MSoslow', 'count': 5140}
{'_id': 'freebeer', 'count': 5107}
{'_id': 'quarkatron', 'count': 4985}
{'_id': 'choess', 'count': 4546}
{'_id': 'ConnectTheDots', 'count': 4478}
{'_id': 'dchiles', 'count': 4340}
{'_id': 'JJeng', 'count': 4200}
{'_id': 'tyos', 'count': 4089}
{'_id': 'peace2', 'count': 3989}
{'_id': '3yoda', 'count': 3606}
{'_id': 'data1', 'count': 3391}
{'_id': 'MTB_Gordon', 'count': 3332}
{'_id': 'aroach', 'count': 3055}
{'_id': 'rivermont', 'count': 3038}
{'_id': 'bhousel', 'count': 3013}
{'_id': 'TIGERcnl', 'count': 2955}
{'_id': 'jjh274', 'count': 2931}
{'_id': 'MastaDLo', 'count': 2631}
{'_id': 'olivad80', 'count': 2558}
{'_id': 'Philip White', 'count': 2432}
{'_id': 'CulingMan13', 'count': 2373}
{'_id': 'Fluous', 'count': 2347}
{'_id': 'kkimtis', 'count': 2216}
{'_id': 'Roadsguy', 'count': 2182}
{'_id': 'trapgodpizza', 'count': 2146}
{'_id': 'crystalwalrein', 'count': 2120}
{'_id': 'bot-mode', 'count': 2053}
{'_id': 'Golfer481977', 'count': 1935}
{'_id': 'TheEnzoBenzo', 'count': 1897}
{'_id': 'TheBestIdea', 'count': 1893}
```

```
{'_id': 'dpawlyk', 'count': 1832}
{'_id': 'Sairam Kotapati', 'count': 1722}
{'_id': 'JKNJ', 'count': 1687}
{'_id': 'PhillyBiker', 'count': 1663}
{'_id': 'pyram', 'count': 1648}
{'_id': 'techiejohn', 'count': 1627}
{'_id': 'JriSv250', 'count': 1600}
{'_id': 'Henry H', 'count': 1570}
{'_id': 'James Michael Dupont (mobile2)', 'count': 1568}
{'_id': 'Victoria1901', 'count': 1538}
{'_id': 'Alex Kuhn', 'count': 1510}
{'_id': 'BigBaby0613', 'count': 1494}
{'_id': 'jaw777', 'count': 1426}
{'_id': 'johnjreiser', 'count': 1396}
{'_id': 'EasternPA', 'count': 1323}
{'_id': 'quincylvania', 'count': 1291}
{'_id': 'soundred', 'count': 1267}
{'_id': 'DressyPear4', 'count': 1217}
{'_id': 'compass26', 'count': 1213}
{'_id': 'sgmamz', 'count': 1210}
{'_id': 'Mavislayer', 'count': 1198}
{'_id': 'mahmedqg', 'count': 1194}
{'_id': 'ASilagyi', 'count': 1180}
{'_id': 'Disodium Inosinate', 'count': 1172}
{'_id': 'natcase', 'count': 1145}
{'_id': 'FitmanNJ', 'count': 1133}
{'_id': 'njtbusfan', 'count': 1112}
{'_id': 'ppallam', 'count': 1087}
{'_id': '42429', 'count': 1019}
{'_id': 'payelgh', 'count': 982}
{'_id': 'RichRico', 'count': 981}
{'_id': 'dvyasha', 'count': 958}
{'_id': 'Riley S', 'count': 952}
{'_id': 'simonbilskyrollins', 'count': 947}
{'_id': 'AsteroidPizza39', 'count': 944}
{'_id': 'pbudhara', 'count': 910}
{'_id': 'iamkrzys', 'count': 896}
{'_id': 'charlesgorby', 'count': 855}
{'_id': 'RD1', 'count': 840}
{'_id': 'dougbrochill', 'count': 836}
{'_id': 'parijatg', 'count': 825}
{'_id': 'rojganes', 'count': 807}
{'_id': 'radioprosciutto', 'count': 776}
{'_id': 'talukdm', 'count': 769}
{'_id': 'nhalat', 'count': 759}
{'_id': 'JimmyRocks', 'count': 740}
{'_id': 'akalohri', 'count': 736}
{'_id': 'Melannial', 'count': 723}
{'_id': 'obom', 'count': 685}
{'_id': 'gohimans', 'count': 668}
{'_id': 'bogdan_andrei', 'count': 660}
{'_id': 'mohapd', 'count': 654}
{'_id': 'Rajsamand Local Guide', 'count': 652}
{'_id': '1T_money', 'count': 645}
{'_id': 'flynnman504', 'count': 632}
{'_id': 'Senalba', 'count': 629}
{'_id': 'sainiyam', 'count': 623}
```

```
{'_id': 'michaedz', 'count': 622}
{'_id': 'mccord42', 'count': 613}
{'_id': 'yerrawa', 'count': 612}
{'_id': 'Fod', 'count': 611}
{'_id': 'justarandomguy07', 'count': 611}
{'_id': 'MartyMartPa', 'count': 601}
{'_id': 'soumybha', 'count': 600}
{'_id': 'bhardwk', 'count': 584}
{'_id': 'karnatim', 'count': 577}
{'_id': 'yadathot', 'count': 573}
{'_id': 'gazsri', 'count': 569}
{'_id': 'maxerickson', 'count': 567}
{'_id': 'kshtdh', 'count': 566}
{'_id': 'ceyockey', 'count': 563}
{'_id': 'DennisL', 'count': 556}
{'_id': 'inagans', 'count': 555}
{'_id': 'vardhamk', 'count': 546}
{'_id': 'ArtofApocalypse', 'count': 545}
{'_id': 'riskhann', 'count': 544}
{'_id': 'ejjoshy', 'count': 536}
{'_id': 'Timothy Smith', 'count': 535}
{'_id': 'clay_c', 'count': 533}
{'_id': 'user_5359', 'count': 527}
{'_id': 'ramesaku', 'count': 527}
{'_id': 'andygol', 'count': 521}
{'_id': 'mcingara', 'count': 514}
{'_id': 'kommsree', 'count': 510}
{'_id': 'ssahithi', 'count': 497}
{'_id': 'mobrien12', 'count': 493}
{'_id': 'srygy', 'count': 486}
{'_id': 'qkkmrg', 'count': 476}
{'_id': 'mubbasa', 'count': 476}
{'_id': 'desuman', 'count': 475}
{'_id': 'vrevanth', 'count': 464}
{'_id': 'shreeush', 'count': 464}
{'_id': 'harshjs', 'count': 464}
{'_id': 'WatchCity', 'count': 461}
{'_id': 'manitec', 'count': 461}
{'_id': 'DoubleA', 'count': 460}
{'_id': 'ommundu', 'count': 450}
{'_id': 'sreejam', 'count': 447}
{'_id': 'bathis', 'count': 444}
{'_id': 'avssr', 'count': 442}
{'_id': 'kmarifo', 'count': 441}
{'_id': 'agnisn', 'count': 440}
{'_id': 'ambarapu', 'count': 439}
{'_id': 'jdoniche', 'count': 439}
{'_id': 'Brandon Gurwitz', 'count': 435}
{'_id': 'kumrrst', 'count': 435}
{'_id': 'jaiaka', 'count': 434}
{'_id': 'KindredCoda', 'count': 432}
{'_id': 'gujelapg', 'count': 432}
{'_id': 'sanayani', 'count': 431}
{'_id': 'rkkomat1', 'count': 429}
{'_id': 'accheela', 'count': 428}
{'_id': 'bakarapu', 'count': 427}
{'_id': 'anabathulaj', 'count': 427}
```

```
{'_id': 'mohmpr', 'count': 420}
{'_id': 'amitbish', 'count': 418}
{'_id': 'cseema', 'count': 416}
{'_id': 'naresksv', 'count': 416}
{'_id': 'kanleela', 'count': 416}
{'_id': 'eadamk', 'count': 413}
{'_id': 'TheAwesomeHwyh', 'count': 413}
{'_id': 'yoasif', 'count': 412}
{'_id': 'rekulc1', 'count': 411}
{'_id': 'venkakaz', 'count': 405}
{'_id': 'nimmalki', 'count': 404}
{'_id': 'pzharsh', 'count': 404}
{'_id': 'jamenimm', 'count': 403}
{'_id': 'Nick Springer', 'count': 398}
{'_id': 'limmmoh', 'count': 389}
{'_id': 'pathkh', 'count': 389}
{'_id': 'kumartcx', 'count': 383}
{'_id': 'vsahith', 'count': 382}
{'_id': 'borvikas', 'count': 381}
{'_id': 'marthaleena', 'count': 381}
{'_id': 'jivins', 'count': 380}
{'_id': 'pandypr', 'count': 378}
{'_id': 'Philly Bike Coalition', 'count': 375}
{'_id': 'sputn1k0ne', 'count': 374}
{'_id': 'iggujja', 'count': 369}
{'_id': 'kotaprad', 'count': 369}
{'_id': 'riysingh', 'count': 368}
{'_id': 'allkrupa', 'count': 368}
{'_id': 'panchis1', 'count': 366}
{'_id': 'ramayang', 'count': 365}
{'_id': 'rppalagu', 'count': 363}
{'_id': 'amzmkuma', 'count': 362}
{'_id': 'xox', 'count': 362}
{'_id': 'sgaraboi', 'count': 361}
{'_id': 'eric22', 'count': 361}
{'_id': 'tejajt', 'count': 361}
{'_id': 'dominastrum', 'count': 360}
{'_id': 'ssange', 'count': 360}
{'_id': 'naniran', 'count': 359}
{'_id': 'rahuzod', 'count': 358}
{'_id': 'poornkum', 'count': 358}
{'_id': 'MathO24', 'count': 353}
{'_id': 'sandchi', 'count': 352}
{'_id': 'chagasru', 'count': 348}
{'_id': 'utkshukl', 'count': 348}
{'_id': 'majumdea', 'count': 346}
{'_id': 'smanidee', 'count': 345}
{'_id': 'sahanki', 'count': 344}
{'_id': 'hyyenugu', 'count': 344}
{'_id': 'sartsuri', 'count': 340}
{'_id': 'dhaird', 'count': 336}
{'_id': 'abidhm', 'count': 335}
{'_id': 'goraia', 'count': 334}
{'_id': 'amillar', 'count': 333}
{'_id': 'nandinab', 'count': 333}
{'_id': 'octants', 'count': 330}
{'_id': 'dggorant', 'count': 329}
```

```
{'_id': 'Nick L', 'count': 329}
{'_id': 'sawhs', 'count': 328}
{'_id': 'modukv', 'count': 328}
{'_id': 'sivaib', 'count': 327}
{'_id': 'samely', 'count': 326}
{'_id': 'shabkhat', 'count': 323}
{'_id': 'sushredd', 'count': 322}
{'_id': 'mvviveka', 'count': 322}
{'_id': 'nihaalr', 'count': 321}
{'_id': 'ntanniru', 'count': 320}
{'_id': 'kirashas', 'count': 314}
{'_id': 'sailajc', 'count': 313}
{'_id': 'dvcphoto92', 'count': 311}
{'_id': 'DarkMagicCat', 'count': 310}
{'_id': 'rrkum', 'count': 310}
{'_id': 'vpprahar', 'count': 307}
{'_id': 'muziriana', 'count': 306}
{'_id': 'srmudava', 'count': 306}
{'_id': 'shoshibu', 'count': 304}
{'_id': 'goingaround', 'count': 304}
{'_id': 'loyakris', 'count': 300}
{'_id': 'fpkerner', 'count': 299}
{'_id': 'qureahme', 'count': 299}
{'_id': 'grandhs', 'count': 299}
{'_id': 'nampeb', 'count': 298}
{'_id': 'anurasi', 'count': 297}
{'_id': 'sun0123', 'count': 296}
{'_id': 'ediyes', 'count': 295}
{'_id': 'Sebastian Hallum Clarke', 'count': 295}
{'_id': 'kllahari', 'count': 294}
{'_id': 'amarajz', 'count': 292}
{'_id': 'scratchspin', 'count': 290}
{'_id': 'dapak', 'count': 290}
{'_id': 'swathiku', 'count': 288}
{'_id': 'jacorohi', 'count': 288}
{'_id': 'kvss', 'count': 284}
{'_id': 'harpreo', 'count': 281}
{'_id': 'devratc', 'count': 279}
{'_id': 'DaveHansenTiger', 'count': 278}
{'_id': 'bhiy', 'count': 276}
{'_id': 'jangah1', 'count': 275}
{'_id': 'mwakram', 'count': 275}
{'_id': 'leelamrs', 'count': 273}
{'_id': 'rzsi', 'count': 271}
{'_id': 'achantav', 'count': 270}
{'_id': 'v0rt3c', 'count': 270}
{'_id': 'indukun', 'count': 266}
{'_id': 'shaspand', 'count': 265}
{'_id': 'baigzake', 'count': 265}
{'_id': 'swarnimv', 'count': 264}
{'_id': 'ahabdu', 'count': 264}
{'_id': 'arkdatta', 'count': 263}
{'_id': 'karragha', 'count': 262}
{'_id': 'pete404', 'count': 262}
{'_id': 'shkshar', 'count': 261}
{'_id': 'flierfy', 'count': 261}
{'_id': 'abel801', 'count': 260}
```

```
{'_id': 'himansne', 'count': 257}
{'_id': 'kaurrupa', 'count': 254}
{'_id': 'bnilima', 'count': 254}
{'_id': 'RoadGeek_MD99', 'count': 254}
{'_id': 'veesams', 'count': 253}
{'_id': 'swimdb', 'count': 253}
{'_id': 'ham', 'count': 252}
{'_id': 'sunilaak', 'count': 252}
{'_id': 'inah_telenav', 'count': 251}
{'_id': 'DannyAiquipa', 'count': 251}
{'_id': 'pbellamk', 'count': 251}
{'_id': 'amithrav', 'count': 251}
{'_id': 'bhavant', 'count': 250}
{'_id': 'silakshm', 'count': 250}
{'_id': 'RussNelson', 'count': 250}
{'_id': 'navlay', 'count': 250}
{'_id': 'unsungNovelty', 'count': 249}
{'_id': 'somaswap', 'count': 246}
{'_id': 'begumfai', 'count': 245}
{'_id': 'rithikaj', 'count': 245}
{'_id': 'AndrewSnow', 'count': 244}
{'_id': 'vatsomya', 'count': 244}
{'_id': 'mohahase', 'count': 244}
{'_id': 'twil4754', 'count': 244}
{'_id': 'riteshaw', 'count': 242}
{'_id': 'AndreasPr', 'count': 241}
{'_id': 'franiu', 'count': 241}
{'_id': 'jssunkar', 'count': 240}
{'_id': 'vissushm', 'count': 239}
{'_id': 'Chetan_Gowda', 'count': 239}
{'_id': 'cat_crash', 'count': 239}
{'_id': 'ashbeesa', 'count': 238}
{'_id': 'chaturvh', 'count': 237}
{'_id': 'srividya_c', 'count': 236}
{'_id': 'vuliset', 'count': 236}
{'_id': 'anilredd', 'count': 234}
{'_id': 'ayushiso', 'count': 234}
{'_id': 'chhava', 'count': 233}
{'_id': 'Bobsus', 'count': 233}
{'_id': 'Valustaides', 'count': 233}
{'_id': 'mitesl', 'count': 233}
{'_id': 'gireeshn', 'count': 232}
{'_id': 'debanka', 'count': 231}
{'_id': 'teliks', 'count': 231}
{'_id': 'asddiqu', 'count': 231}
{'_id': 'ssomeya', 'count': 230}
{'_id': 'chekasat', 'count': 229}
{'_id': 'syrah1', 'count': 229}
{'_id': 'paliks', 'count': 228}
{'_id': 'vbbukka', 'count': 227}
{'_id': 'scharith', 'count': 227}
{'_id': 'debott', 'count': 226}
{'_id': 'Luis36995', 'count': 225}
{'_id': 'boddushr', 'count': 224}
{'_id': 'Constable', 'count': 223}
{'_id': 'kurugunk', 'count': 222}
{'_id': 'vchndl', 'count': 220}
```

```
{'_id': 'gdodlapa', 'count': 219}
{'_id': 'purbitap', 'count': 218}
{'_id': 'govvala', 'count': 217}
{'_id': 'skharwal', 'count': 217}
{'_id': 'Taknamay', 'count': 216}
{'_id': 'ToffeHoff', 'count': 216}
{'_id': 'utkap', 'count': 216}
{'_id': 'nllucky', 'count': 215}
{'_id': 'kedarnap', 'count': 215}
{'_id': 'nschg', 'count': 214}
{'_id': 'SophoM', 'count': 214}
{'_id': 'tmadhuli', 'count': 213}
{'_id': 'mrfa', 'count': 211}
{'_id': 'Vkkalegi', 'count': 210}
{'_id': 'siddhaam', 'count': 210}
{'_id': 'abhap', 'count': 210}
{'_id': 'CurlingMan13', 'count': 207}
{'_id': 'avipsha', 'count': 207}
{'_id': 'poornibadrinath', 'count': 204}
{'_id': 'rkkasams', 'count': 203}
{'_id': 'Polarbear', 'count': 203}
{'_id': 'zmankits', 'count': 201}
{'_id': 'pupsachi', 'count': 201}
{'_id': 'shypanda', 'count': 201}
{'_id': 'muddnnz', 'count': 199}
{'_id': 'sabaamzn', 'count': 198}
{'_id': 'John the Mapper', 'count': 197}
{'_id': 'dasnei', 'count': 196}
{'_id': 'tejkante', 'count': 196}
{'_id': 'pillalp', 'count': 195}
{'_id': 'calfarome', 'count': 194}
{'_id': 'pmmegh', 'count': 193}
{'_id': 'pirigine', 'count': 191}
{'_id': 'vddontha', 'count': 191}
{'_id': 'rapush', 'count': 190}
{'_id': 'ganarend', 'count': 188}
{'_id': 'nbhiss', 'count': 188}
{'_id': 'dyellak', 'count': 187}
{'_id': 'arosalon', 'count': 187}
{'_id': 'manapra', 'count': 186}
{'_id': 'MGSmithNJ', 'count': 186}
{'_id': 'rkkarnee', 'count': 185}
{'_id': 'nishisi', 'count': 185}
{'_id': 'devapujk', 'count': 185}
{'_id': 'mbonte', 'count': 183}
{'_id': 'jedidy', 'count': 183}
{'_id': 'dlu306090', 'count': 182}
{'_id': 'Tim Perpedic', 'count': 180}
{'_id': 'patadivy', 'count': 179}
{'_id': 'sanganh', 'count': 178}
{'_id': 'raonara', 'count': 177}
{'_id': 'jomsjaco', 'count': 177}
{'_id': 'vchandra007', 'count': 177}
{'_id': 'mvexel', 'count': 176}
{'_id': 'MCollar', 'count': 176}
{'_id': 'duggemps', 'count': 175}
{'_id': 'dale_p', 'count': 174}
```

```
{'_id': 'chatrags', 'count': 174}
{'_id': 'vvvuppal', 'count': 173}
{'_id': 'chosayan', 'count': 173}
{'_id': 'Prof Eric Cameron', 'count': 173}
{'_id': 'vkanduku', 'count': 173}
{'_id': 'pkkv', 'count': 173}
{'_id': 'ajashton', 'count': 172}
{'_id': 'nallivn', 'count': 171}
{'_id': 'reswara', 'count': 171}
{'_id': 'ajacobi5', 'count': 171}
{'_id': 'ppjj', 'count': 169}
{'_id': 'tahuram', 'count': 169}
{'_id': 'hkkollip', 'count': 169}
{'_id': 'posrujan', 'count': 168}
{'_id': 'duarisha', 'count': 168}
{'_id': 'Reemal', 'count': 168}
{'_id': 'bjnlm', 'count': 167}
{'_id': 'jyotkuma', 'count': 165}
{'_id': 'pantus', 'count': 165}
{'_id': 'jhameena', 'count': 164}
{'_id': 'kirrawat', 'count': 164}
{'_id': 'mandalat', 'count': 164}
{'_id': 'bhousel_bot', 'count': 164}
{'_id': 'devasen', 'count': 163}
{'_id': 'mhdrahm', 'count': 163}
{'_id': 'hemasvn', 'count': 163}
{'_id': 'ammhita', 'count': 161}
{'_id': 'andrewpmk', 'count': 160}
{'_id': 'gilasri', 'count': 159}
{'_id': 'Kevin_J', 'count': 159}
{'_id': 'paasthan', 'count': 158}
{'_id': 'tusnayak', 'count': 157}
{'_id': 'knikitha', 'count': 157}
{'_id': 'shargrac', 'count': 157}
{'_id': 'NW757', 'count': 155}
{'_id': 'w9ddVKFwXt7KqWmj', 'count': 154}
{'_id': 'nsripz', 'count': 154}
{'_id': 'sgudiva', 'count': 153}
{'_id': 'nambim', 'count': 153}
{'_id': 'David Meni', 'count': 153}
{'_id': 'daggarw', 'count': 153}
{'_id': 'pppadal', 'count': 151}
{'_id': 'Utsab Khatiwada', 'count': 151}
{'_id': 'buchula', 'count': 151}
{'_id': 'sivapaid', 'count': 151}
{'_id': 'ksrilekh', 'count': 150}
{'_id': 'naaitha', 'count': 149}
{'_id': 'vysyarp', 'count': 149}
{'_id': 'Ddoolittle86', 'count': 149}
{'_id': 'Rupert Swarbrick', 'count': 148}
{'_id': 'anjefu', 'count': 147}
{'_id': 'jojoyal', 'count': 147}
{'_id': 'stoyrahu', 'count': 146}
{'_id': 'S-zation', 'count': 146}
{'_id': 'johnnymapp', 'count': 144}
{'_id': 'adhya1', 'count': 144}
{'_id': 'ionutr_telenav', 'count': 143}
```

```
{'_id': 'svvakkal', 'count': 142}
{'_id': 'nimmalab', 'count': 142}
{'_id': 'shadty', 'count': 142}
{'_id': 'runkanp', 'count': 141}
{'_id': 'mathpras', 'count': 140}
{'_id': 'harishwr', 'count': 139}
{'_id': 'iandees', 'count': 139}
{'_id': 'mapalla', 'count': 139}
{'_id': 'prafupan', 'count': 139}
{'_id': 'vvvnm', 'count': 139}
{'_id': 'sanuhya', 'count': 137}
{'_id': 'rppalle', 'count': 137}
{'_id': 'sairamns', 'count': 136}
{'_id': 'himabudd', 'count': 136}
{'_id': 'shushmit', 'count': 136}
{'_id': 'hofoen', 'count': 135}
{'_id': 'snyash', 'count': 134}
{'_id': 'abvincen', 'count': 133}
{'_id': 'mashakum', 'count': 133}
{'_id': 'sabok', 'count': 133}
{'_id': 'almaasm', 'count': 132}
{'_id': 'suhebm', 'count': 132}
{'_id': 'kilarv', 'count': 132}
{'_id': 'greggerm', 'count': 132}
{'_id': 'Yoshinion', 'count': 132}
{'_id': 'steverumizen', 'count': 132}
{'_id': 'kkre1', 'count': 131}
{'_id': 'arehaan', 'count': 129}
{'_id': 'Rub21', 'count': 128}
{'_id': 'srirohan', 'count': 128}
{'_id': 'cswaroo', 'count': 127}
{'_id': 'racranji', 'count': 126}
{'_id': 'nemodio', 'count': 126}
{'_id': 'aji8ful', 'count': 126}
{'_id': 'gollabg', 'count': 125}
{'_id': 'chandeek', 'count': 124}
{'_id': 'bahnpirat', 'count': 124}
{'_id': 'dppola', 'count': 124}
{'_id': 'EvanSiroky', 'count': 123}
{'_id': 'saikariv', 'count': 123}
{'_id': 'teodorab_telenav', 'count': 122}
{'_id': 'bapup', 'count': 121}
{'_id': 'ramijcr', 'count': 120}
{'_id': 'SublimeDutch', 'count': 120}
{'_id': 'Andrew Maiman', 'count': 120}
{'_id': 'vshnsa', 'count': 119}
{'_id': 'vvrddy', 'count': 119}
{'_id': 'yallasan', 'count': 119}
{'_id': 'sskalyan', 'count': 119}
{'_id': 'jalajm', 'count': 118}
{'_id': 'sivachak', 'count': 117}
{'_id': 'ekkakkas', 'count': 117}
{'_id': 'gundanar', 'count': 116}
{'_id': 'pdantojh', 'count': 116}
{'_id': 'vaidhev', 'count': 115}
{'_id': 'tammpava', 'count': 115}
{'_id': 'navuddin', 'count': 114}
```

```
{'_id': 'mcld', 'count': 114}
{'_id': 'Luisi Mouse', 'count': 113}
{'_id': 'saksudan', 'count': 113}
{'_id': 'attaragi', 'count': 112}
{'_id': 'swarood', 'count': 112}
{'_id': 'bachurev', 'count': 112}
{'_id': 'varmasa', 'count': 112}
{'_id': 'thaggela', 'count': 111}
{'_id': 'vkatredd', 'count': 110}
{'_id': '25or6to4', 'count': 110}
{'_id': 'penimire', 'count': 110}
{'_id': 'shadijan', 'count': 110}
{'_id': 'rajathiraja', 'count': 110}
{'_id': 'varshia', 'count': 109}
{'_id': 'bmmh', 'count': 109}
{'_id': 'hpeddir', 'count': 108}
{'_id': 'Map Corrections', 'count': 108}
{'_id': 'anjohn12', 'count': 107}
{'_id': 'bvvams', 'count': 106}
{'_id': 'rstb', 'count': 105}
{'_id': 'sankethn', 'count': 103}
{'_id': 'deepikja', 'count': 103}
{'_id': 'drewmhess', 'count': 102}
{'_id': 'Waymom', 'count': 102}
{'_id': 'jonesydesign', 'count': 102}
{'_id': 'Edward', 'count': 102}
{'_id': 'FonBaron', 'count': 101}
{'_id': 'MikeN', 'count': 101}
{'_id': 'swappa', 'count': 101}
{'_id': 'dviraja', 'count': 101}
{'_id': 'kavshnu', 'count': 100}
{'_id': 'nairnidh', 'count': 99}
{'_id': 'DaytwanCH', 'count': 99}
{'_id': 'kevmath', 'count': 98}
{'_id': 'OSchlüter', 'count': 97}
{'_id': 'gauraan', 'count': 97}
{'_id': 'krjayakr', 'count': 96}
{'_id': 'L Frcht', 'count': 96}
{'_id': 'mansipan', 'count': 96}
{'_id': 'papankum', 'count': 96}
{'_id': 'Cliffsilo', 'count': 96}
{'_id': 'saxvidit', 'count': 95}
{'_id': 'Chris Knigge', 'count': 95}
{'_id': 'latkins1', 'count': 95}
{'_id': 'Jasmine2009', 'count': 95}
{'_id': 'markzawi', 'count': 93}
{'_id': 'somanch', 'count': 93}
{'_id': 'sunaman', 'count': 93}
{'_id': 'saikabhi', 'count': 92}
{'_id': 'praghath', 'count': 92}
{'_id': 'asrdd', 'count': 92}
{'_id': 'cforin', 'count': 91}
{'_id': 'flevin', 'count': 91}
{'_id': 'amznruth', 'count': 91}
{'_id': 'John Boyle', 'count': 91}
{'_id': 'ChrisZontine', 'count': 91}
{'_id': 'saisidd', 'count': 90}
```

```
{'_id': 'sahithpe', 'count': 89}
{'_id': 'Rondale', 'count': 89}
{'_id': 'manikyal', 'count': 87}
{'_id': 'nehnaaz', 'count': 87}
{'_id': 'antoalvi', 'count': 86}
{'_id': 'pachpora', 'count': 86}
{'_id': 'pwolanin', 'count': 86}
{'_id': 'boopington', 'count': 86}
{'_id': 'arpremna', 'count': 85}
{'_id': 'krishcss', 'count': 83}
{'_id': 'heyimvictor', 'count': 83}
{'_id': 'AveryDorgan', 'count': 83}
{'_id': 'JessAk71', 'count': 83}
{'_id': 'rekasruj', 'count': 83}
{'_id': 'svarkey', 'count': 82}
{'_id': 'jonsger', 'count': 82}
{'_id': 'ppatlmn', 'count': 81}
{'_id': 'egore911', 'count': 81}
{'_id': 'chepteja', 'count': 81}
{'_id': 'mzamam', 'count': 81}
{'_id': 'raknm', 'count': 80}
{'_id': 'ECRock', 'count': 80}
{'_id': 'granpueblo', 'count': 80}
{'_id': 'vjjeevan', 'count': 80}
{'_id': 'gauvibhu', 'count': 79}
{'_id': 'manoharuss', 'count': 79}
{'_id': 'cvamsi', 'count': 78}
{'_id': 'Iowa Kid', 'count': 78}
{'_id': 'Ribechan', 'count': 78}
{'_id': 'sireeshp', 'count': 77}
{'_id': 'matinamongo', 'count': 77}
{'_id': 'ntilley905', 'count': 77}
{'_id': 'uhhany1', 'count': 77}
{'_id': 'panmohin', 'count': 77}
{'_id': 'WhiteRakogis', 'count': 77}
{'_id': 'ashradha', 'count': 77}
{'_id': 'Hamza Roadbotics', 'count': 76}
{'_id': 'EdSS', 'count': 76}
{'_id': 'manasaka', 'count': 76}
{'_id': 'buddhirv', 'count': 75}
{'_id': 'deanabil', 'count': 75}
{'_id': 'sbbollep', 'count': 75}
{'_id': 'nadikoh', 'count': 74}
{'_id': 'Yojoe73', 'count': 74}
{'_id': 'javmoh', 'count': 74}
{'_id': 'sachinv', 'count': 73}
{'_id': 'eringreb', 'count': 73}
{'_id': 'kkoganti', 'count': 73}
{'_id': 'pillaap', 'count': 73}
{'_id': 'James Baxter', 'count': 73}
{'_id': 'Parcanman', 'count': 73}
{'_id': 'gabis_telenav', 'count': 73}
{'_id': 'ksnghr', 'count': 72}
{'_id': 'di5order', 'count': 72}
{'_id': 'vasamd', 'count': 72}
{'_id': 'Carldc', 'count': 72}
{'_id': 'gandlur', 'count': 72}
```

```
{'_id': 'IanH', 'count': 72}
{'_id': 'uboot', 'count': 72}
{'_id': 'Aleksei Potov', 'count': 72}
{'_id': 'floverde', 'count': 72}
{'_id': 'Dom007', 'count': 71}
{'_id': 'mubshh', 'count': 71}
{'_id': 'mramkr', 'count': 71}
{'_id': 'StaffMapEditor', 'count': 71}
{'_id': 'KristenK', 'count': 70}
{'_id': 'vkkasarl', 'count': 69}
{'_id': 'Bhojaraj', 'count': 69}
{'_id': 'AGED P', 'count': 69}
{'_id': 'snghcvv', 'count': 68}
{'_id': 'kmarkish_lyft', 'count': 68}
{'_id': 'Tom100', 'count': 67}
{'_id': 'bingo', 'count': 67}
{'_id': 'ygudeti', 'count': 67}
{'_id': 'Mundilfari', 'count': 67}
{'_id': 'dhakanth', 'count': 67}
{'_id': 'shasband', 'count': 66}
{'_id': 'Claumires', 'count': 66}
{'_id': 'miluethi', 'count': 65}
{'_id': 'emiliovigil826', 'count': 65}
{'_id': 'chettups', 'count': 64}
{'_id': 'Maskulinum', 'count': 64}
{'_id': 'karitotp', 'count': 63}
{'_id': 'MajorTC', 'count': 63}
{'_id': 'sinreeya', 'count': 63}
{'_id': 'Sphinxiro', 'count': 63}
{'_id': 'ridixcr', 'count': 62}
{'_id': 'dasthagd', 'count': 62}
{'_id': 'kfremd', 'count': 62}
{'_id': 'jshaju', 'count': 62}
{'_id': 'herriotto', 'count': 62}
{'_id': 'rickmastfan67', 'count': 61}
{'_id': 'NJDOT FPEC', 'count': 61}
{'_id': 'yaswap', 'count': 61}
{'_id': 'andreis_telenav', 'count': 61}
{'_id': 'mshahaba', 'count': 60}
{'_id': 'harsmula', 'count': 60}
{'_id': 'anuvashi', 'count': 60}
{'_id': 'Takuto', 'count': 60}
{'_id': '1248', 'count': 59}
{'_id': 'Man89090', 'count': 59}
{'_id': 'fordelit', 'count': 59}
{'_id': 'rayaraoa', 'count': 59}
{'_id': 'pareps', 'count': 59}
{'_id': 'pagibson', 'count': 59}
{'_id': 'disjoinedking', 'count': 58}
{'_id': 'khnzmnz', 'count': 58}
{'_id': 'bnewbold', 'count': 58}
{'_id': 'malihars', 'count': 58}
{'_id': 'makkera', 'count': 58}
{'_id': 'Upendrakarukonda', 'count': 58}
{'_id': 'DavidLiu5059', 'count': 57}
{'_id': 'Nat Bradley', 'count': 57}
{'_id': 'becw', 'count': 57}
```

```
{'_id': 'kotharu', 'count': 56}
{'_id': 'kollipay', 'count': 56}
{'_id': 'rddeepth', 'count': 56}
{'_id': 'vmonisha', 'count': 56}
{'_id': 'Dion Dock', 'count': 55}
{'_id': 'niraredd', 'count': 55}
{'_id': 'WesWeaver', 'count': 55}
{'_id': 'ryandrake', 'count': 55}
{'_id': 'karsonkevin2', 'count': 54}
{'_id': 'thetornado76', 'count': 54}
{'_id': 'dcat', 'count': 54}
{'_id': 'FTA', 'count': 53}
{'_id': 'muksoham', 'count': 53}
{'_id': 'VLD085', 'count': 52}
{'_id': 'gavra', 'count': 52}
{'_id': 'jumbanho', 'count': 52}
{'_id': 'sanbukke', 'count': 52}
{'_id': 'Spanholz', 'count': 52}
{'_id': 'simonap_telenav', 'count': 52}
{'_id': 'FvGordon', 'count': 52}
{'_id': 'b-jazz-bot', 'count': 51}
{'_id': 'srinatpa', 'count': 50}
{'_id': 'lelliloo', 'count': 49}
{'_id': 'ksd5', 'count': 49}
{'_id': 'uthkarsh1', 'count': 49}
{'_id': 'maitsana', 'count': 49}
{'_id': 'anuds', 'count': 49}
{'_id': 'shishf', 'count': 49}
{'_id': 'janicea', 'count': 49}
{'_id': 'kurtkcpeng', 'count': 48}
{'_id': 'Jason Harrison', 'count': 48}
{'_id': 'LizF', 'count': 48}
{'_id': 'Andrew Matheny', 'count': 48}
{'_id': 'roshanparajuli', 'count': 48}
{'_id': 'pramvyas', 'count': 47}
{'_id': 'Klaus Schwer', 'count': 47}
{'_id': 'smbharan', 'count': 47}
{'_id': 'nguyent10', 'count': 47}
{'_id': 'kartonage', 'count': 46}
{'_id': 'pddondet', 'count': 46}
{'_id': 'georges_telenav', 'count': 45}
{'_id': 'amakarevich_lyft', 'count': 45}
{'_id': 'VLD151', 'count': 45}
{'_id': 'aepunavy', 'count': 45}
{'_id': 'RationalTangle', 'count': 45}
{'_id': 'pattamatta', 'count': 44}
{'_id': 'mudunur', 'count': 44}
{'_id': 'manings', 'count': 44}
{'_id': 'maakra', 'count': 44}
{'_id': 'baindran', 'count': 43}
{'_id': 'chharsha', 'count': 43}
{'_id': 'ioanam_telenav', 'count': 43}
{'_id': 'pratikyadav', 'count': 42}
{'_id': 'krinkov76239', 'count': 42}
{'_id': 'sggundl', 'count': 41}
{'_id': 'Polyglot', 'count': 41}
{'_id': 'Hello098You', 'count': 41}
```

```
{'_id': 'osm-sputnik', 'count': 41}
{'_id': 'b-jazz', 'count': 41}
{'_id': 'psusmith', 'count': 40}
{'_id': 'posampat', 'count': 39}
{'_id': 'kararcha', 'count': 38}
{'_id': 'hoream_telenav', 'count': 38}
{'_id': 'abbum', 'count': 38}
{'_id': 'irasesu', 'count': 38}
{'_id': 'gadipeh', 'count': 38}
{'_id': 'gsravya1', 'count': 37}
{'_id': 'A-F', 'count': 36}
{'_id': 'wvdp', 'count': 36}
{'_id': 'kmarjbh', 'count': 36}
{'_id': 'Risho Kanthala', 'count': 36}
{'_id': 'savithrd', 'count': 36}
{'_id': 'Bridger', 'count': 35}
{'_id': 'hannah32', 'count': 35}
{'_id': 'spavulur', 'count': 35}
{'_id': 'Omar Nishtar', 'count': 35}
{'_id': 'someart13', 'count': 35}
{'_id': 'mdnade', 'count': 35}
{'_id': 'BHalloran', 'count': 35}
{'_id': 'beweta', 'count': 35}
{'_id': 'enaumov', 'count': 35}
{'_id': 'galithot', 'count': 35}
{'_id': 'airangam', 'count': 35}
{'_id': 'kmneh', 'count': 34}
{'_id': 'MBedilion', 'count': 34}
{'_id': 'mttgrmm', 'count': 33}
{'_id': 'akakhno_lyft', 'count': 33}
{'_id': 'bhaktano1', 'count': 33}
{'_id': 'VLD071', 'count': 33}
{'_id': 'Omnific', 'count': 33}
{'_id': 'edack', 'count': 33}
{'_id': 'murd0ch', 'count': 33}
{'_id': 'GCEA08', 'count': 33}
{'_id': 'rbhawa', 'count': 32}
{'_id': 'rsakunth', 'count': 32}
{'_id': 'Robopuppy1992', 'count': 32}
{'_id': 'Xhris', 'count': 32}
{'_id': 'felipeeugenio', 'count': 32}
{'_id': 'yurasi', 'count': 32}
{'_id': 'edathy', 'count': 31}
{'_id': 'mapper611', 'count': 31}
{'_id': 'mhe500', 'count': 31}
{'_id': 'Lika87', 'count': 31}
{'_id': 'befit1', 'count': 31}
{'_id': 'kentycheng', 'count': 31}
{'_id': 'SlowbroArmor094', 'count': 31}
{'_id': 'Aidan Snow', 'count': 31}
{'_id': 'rpakma', 'count': 31}
{'_id': 'Bhimesh varyani', 'count': 31}
{'_id': 'MadBrad', 'count': 30}
{'_id': '110110110101011110', 'count': 30}
{'_id': 'neuhausr', 'count': 30}
{'_id': 'vy1587', 'count': 29}
{'_id': 'DzmitryBachkarou', 'count': 29}
```

```
{'_id': 'paulinelv', 'count': 29}
{'_id': 'vaisn', 'count': 29}
{'_id': 'Teesta', 'count': 29}
{'_id': 'sreelekk', 'count': 28}
{'_id': 'Blimeo', 'count': 28}
{'_id': 'gouldpr0', 'count': 28}
{'_id': 'Monica Brandeis', 'count': 28}
{'_id': 'oormilavinod', 'count': 28}
{'_id': 'vrynkevich_lyft', 'count': 27}
{'_id': 'katakp', 'count': 27}
{'_id': 'stadiaarcadia', 'count': 27}
{'_id': 'Gabriel Ehrnst Grundin', 'count': 27}
{'_id': 'bsreeja', 'count': 27}
{'_id': 'skquinn', 'count': 27}
{'_id': 'Jowzjan', 'count': 27}
{'_id': 'abboddul', 'count': 27}
{'_id': 'gpotato', 'count': 27}
{'_id': 'kmitshu', 'count': 27}
{'_id': 'emremre', 'count': 26}
{'_id': 'kanthvk', 'count': 26}
{'_id': 'cma2', 'count': 26}
{'_id': 'Brad Murray', 'count': 26}
{'_id': 'bhavana naga', 'count': 26}
{'_id': 'afj3278d', 'count': 26}
{'_id': 'paulCIA', 'count': 25}
{'_id': 'pbobbili', 'count': 25}
{'_id': 'Fa7C0N', 'count': 25}
{'_id': 'TLJasonWyatt', 'count': 25}
{'_id': 'BlindWanderer', 'count': 25}
{'_id': 'florinbadita_telenav', 'count': 25}
{'_id': 'dondiego87', 'count': 25}
{'_id': 'nammala', 'count': 25}
{'_id': 'aggopak', 'count': 25}
{'_id': 'singhyyd', 'count': 25}
{'_id': 'dupputur', 'count': 24}
{'_id': 'ammendir', 'count': 24}
{'_id': 'VitaminD', 'count': 24}
{'_id': 'technician_lyft', 'count': 23}
{'_id': 'azapf1972', 'count': 23}
{'_id': 'lukasmartinelli', 'count': 23}
{'_id': 'aighes', 'count': 23}
{'_id': 'hi_there', 'count': 23}
{'_id': '123458821', 'count': 23}
{'_id': 'dasbham', 'count': 23}
{'_id': 'prachen', 'count': 23}
{'_id': 'stevea', 'count': 23}
{'_id': 'VLD160', 'count': 22}
{'_id': 'nurban', 'count': 22}
{'_id': 'qsudarji', 'count': 22}
{'_id': 'RDaneel', 'count': 22}
{'_id': 'Imp_GL', 'count': 22}
{'_id': 'VLD174', 'count': 22}
{'_id': 'Fluffy89502', 'count': 22}
{'_id': 'moalnzn', 'count': 22}
{'_id': 'MapSpot', 'count': 22}
{'_id': 'SuperDuperHerb', 'count': 22}
{'_id': 'gate63', 'count': 22}
```

```
{'_id': 'Baloo Uriza', 'count': 22}
{'_id': 'vinitht', 'count': 21}
{'_id': 'annajyoung', 'count': 21}
{'_id': 'kpcunnin', 'count': 21}
{'_id': 'a98alvin', 'count': 21}
{'_id': 'VLD165', 'count': 21}
{'_id': 'djo_man', 'count': 21}
{'_id': 'Firemix', 'count': 21}
{'_id': 'Briik', 'count': 21}
{'_id': 'deadoth', 'count': 21}
{'_id': 'Mapman391', 'count': 21}
{'_id': 'wegavision', 'count': 21}
{'_id': 'Konda here', 'count': 21}
{'_id': '!rf@n', 'count': 20}
{'_id': 'Mike Coles', 'count': 20}
{'_id': 'addatla', 'count': 20}
{'_id': 'manognar', 'count': 20}
{'_id': 'OklaNHD', 'count': 20}
{'_id': 'Calophi', 'count': 20}
{'_id': 'srigirip', 'count': 20}
{'_id': 'smw87', 'count': 20}
{'_id': 'wen_li', 'count': 20}
{'_id': 'fx99', 'count': 19}
{'_id': 'Cato_d_Ae', 'count': 19}
{'_id': 'Geogast', 'count': 19}
{'_id': 'oanac2_telenav', 'count': 19}
{'_id': 'CaitlinMOpenStreetMap', 'count': 19}
{'_id': 'MountJoyPA', 'count': 19}
{'_id': 'YuliyaShustava_lyft', 'count': 19}
{'_id': 'Phil Scherer', 'count': 19}
{'_id': 'Goffredo', 'count': 19}
{'_id': 'rkumreo', 'count': 19}
{'_id': 'mtmail', 'count': 18}
{'_id': 'lsshruti', 'count': 18}
{'_id': 'houston_mapper1', 'count': 18}
{'_id': 'Osmosefixer2021', 'count': 18}
{'_id': 'vishsath', 'count': 18}
{'_id': 'dpascual', 'count': 18}
{'_id': 'Atanas Entchev', 'count': 18}
{'_id': 'bwarren', 'count': 17}
{'_id': 'Milenko', 'count': 17}
{'_id': 'sai123003', 'count': 17}
{'_id': 'pa225099', 'count': 17}
{'_id': 'skodeboi', 'count': 17}
{'_id': 'lorandr_telenav', 'count': 17}
{'_id': 'unnsyeda', 'count': 17}
{'_id': 'jazztunes', 'count': 17}
{'_id': 'venkanna37', 'count': 17}
{'_id': 'sruthp', 'count': 16}
{'_id': 'pshalyt_lyft', 'count': 16}
{'_id': 'ravsjith', 'count': 16}
{'_id': 'madharor', 'count': 16}
{'_id': 'Pookaguy', 'count': 16}
{'_id': 'Iqhra', 'count': 16}
{'_id': 'TeresaPeteti', 'count': 16}
{'_id': 'DavidF', 'count': 16}
{'_id': 'amyb', 'count': 15}
```

```
{'_id': 'motios', 'count': 15}
{'_id': 'DianaDemchuk', 'count': 15}
{'_id': 'sunilvs', 'count': 15}
{'_id': 'yerramg', 'count': 15}
{'_id': 'datbrahm', 'count': 15}
{'_id': 'ChrisK123', 'count': 15}
{'_id': 'rdanouski_lyft', 'count': 15}
{'_id': 'rnupam', 'count': 15}
{'_id': 'alester', 'count': 15}
{'_id': 'mueschel', 'count': 14}
{'_id': 'an engineer', 'count': 14}
{'_id': 'ychatur', 'count': 14}
{'_id': 'cranrob', 'count': 14}
{'_id': 'tws45', 'count': 14}
{'_id': 'dt65', 'count': 14}
{'_id': 'ppgne', 'count': 14}
{'_id': 'nickdv', 'count': 13}
{'_id': 'ukalpriy', 'count': 13}
{'_id': 'nmotupal', 'count': 13}
{'_id': 'A_Prokopova_lyft', 'count': 13}
{'_id': 'Mark Bastian', 'count': 13}
{'_id': 'Sugbu', 'count': 13}
{'_id': 'piligab', 'count': 13}
{'_id': 'Geekly', 'count': 13}
{'_id': 'Bonya_23', 'count': 13}
{'_id': 'Jorisbo', 'count': 13}
{'_id': 'spatrick467', 'count': 13}
{'_id': 'gibberflibber', 'count': 13}
{'_id': 'Adamant1', 'count': 12}
{'_id': 'pmarkina_lyft', 'count': 12}
{'_id': 'MochiNJ', 'count': 12}
{'_id': 'matty339', 'count': 12}
{'_id': 'Vlad', 'count': 12}
{'_id': 'nkhall', 'count': 12}
{'_id': 'ashishub', 'count': 12}
{'_id': 'Varun Talwar', 'count': 12}
{'_id': 'Dowluri', 'count': 12}
{'_id': 'TheSwavu', 'count': 12}
{'_id': 'emacsen', 'count': 12}
{'_id': 'Christoph Lotz', 'count': 12}
{'_id': 'stevethecoolguy', 'count': 12}
{'_id': 'BMACS001', 'count': 12}
{'_id': 'TMLutas', 'count': 12}
{'_id': 'awethor', 'count': 12}
{'_id': 'TheRoadDudeMN', 'count': 12}
{'_id': 'alixhartmann', 'count': 12}
{'_id': 'christopherwrong', 'count': 12}
{'_id': 'fredjunod', 'count': 12}
{'_id': 'Lipskylip', 'count': 12}
{'_id': 'laflovver', 'count': 11}
{'_id': 'lindseycoleman', 'count': 11}
{'_id': 'ppparsi', 'count': 11}
{'_id': 'marc lefebvre', 'count': 11}
{'_id': 'appank', 'count': 11}
{'_id': 'pravallg', 'count': 11}
{'_id': 'jaisamya', 'count': 11}
{'_id': 'vladimir_gnedko', 'count': 11}
```

```
{'_id': 'robgeb', 'count': 11}
{'_id': 'vermiceli', 'count': 11}
{'_id': 'muthyamj', 'count': 10}
{'_id': 'Heshy', 'count': 10}
{'_id': 'musaib ali', 'count': 10}
{'_id': 'Doug Coen', 'count': 10}
{'_id': 'KanaLee', 'count': 10}
{'_id': 'mavudur', 'count': 10}
{'_id': 'ZeLonewolf', 'count': 10}
{'_id': 'JPClement', 'count': 10}
{'_id': 'thisischuck01', 'count': 10}
{'_id': 'PHsiao', 'count': 10}
{'_id': 'VLD170', 'count': 10}
{'_id': 'laurenapp', 'count': 10}
{'_id': 'CJenson', 'count': 10}
{'_id': 'cpalanki', 'count': 10}
{'_id': 'Adler777', 'count': 9}
{'_id': 'siayu', 'count': 9}
{'_id': 'dlapo_lyft', 'count': 9}
{'_id': 'Wolcott Blair', 'count': 9}
{'_id': 'sagisa', 'count': 9}
{'_id': 'volite', 'count': 9}
{'_id': 'ashleyannmathew', 'count': 9}
{'_id': 'IDSRichyC', 'count': 9}
{'_id': 'hnlohrey', 'count': 9}
{'_id': 'greenorangeparrot', 'count': 9}
{'_id': 'skwidney96', 'count': 9}
{'_id': 'rrnju', 'count': 9}
{'_id': 'hgy', 'count': 9}
{'_id': 'Andre68', 'count': 9}
{'_id': 'Phil Zhang', 'count': 9}
{'_id': 'LennyPak_lyft', 'count': 9}
{'_id': 'RaquelFish09', 'count': 9}
{'_id': 'mrbarker', 'count': 9}
{'_id': 'S-hornbaker', 'count': 9}
{'_id': 'VLD161', 'count': 9}
{'_id': 'illia ivanou', 'count': 9}
{'_id': 'VLD164', 'count': 9}
{'_id': 'Md Alamgir', 'count': 9}
{'_id': 'A_Youssef', 'count': 8}
{'_id': 'Sammyhawkrad', 'count': 8}
{'_id': 'nevw', 'count': 8}
{'_id': 'Geofreund1', 'count': 8}
{'_id': 'JSp1152', 'count': 8}
{'_id': 'nm7s9', 'count': 8}
{'_id': 'Erin Sheets', 'count': 8}
{'_id': 'smediump', 'count': 8}
{'_id': 'nrohsing', 'count': 8}
{'_id': 'R_U', 'count': 8}
{'_id': 'MilaZ', 'count': 8}
{'_id': 'ammrita', 'count': 8}
{'_id': 'NastassiaKalesnikava', 'count': 8}
{'_id': 'rmmunnan', 'count': 8}
{'_id': 'SomeoneElse_Revert', 'count': 8}
{'_id': 'changurl', 'count': 8}
{'_id': 'Pest Wednesday', 'count': 8}
{'_id': 'Benoît Prieur', 'count': 7}
```

```
{'_id': 'piyuku', 'count': 7}
{'_id': 'VLD176', 'count': 7}
{'_id': 'rad1ance', 'count': 7}
{'_id': 'T_9er', 'count': 7}
{'_id': 'Elinorzag', 'count': 7}
{'_id': 'Janjko', 'count': 7}
{'_id': 'Mashin', 'count': 7}
{'_id': 'PSUFan12', 'count': 7}
{'_id': 'Mert Peksen', 'count': 7}
{'_id': 'twc48', 'count': 7}
{'_id': 'bashwith', 'count': 7}
{'_id': 'Kyrgyzstan345', 'count': 7}
{'_id': 'James Tomasino', 'count': 7}
{'_id': 'trigpoint', 'count': 7}
{'_id': 'Thepilo', 'count': 7}
{'_id': 'Richard', 'count': 7}
{'_id': 'TheHammer', 'count': 7}
{'_id': 'mdevinen', 'count': 7}
{'_id': 'laharicl', 'count': 7}
{'_id': 'Slava Andreyev', 'count': 7}
{'_id': 'pattamaa', 'count': 7}
{'_id': 'Rickyrab1', 'count': 6}
{'_id': 'DanX', 'count': 6}
{'_id': 'wheelmap_visitor', 'count': 6}
{'_id': 'Supernovava', 'count': 6}
{'_id': 'map93', 'count': 6}
{'_id': 'miuddin', 'count': 6}
{'_id': 'Dmithry', 'count': 6}
{'_id': 'YTaulai_lyft', 'count': 6}
{'_id': 'sijjacob', 'count': 6}
{'_id': 'amahon', 'count': 6}
{'_id': 'acttest', 'count': 6}
{'_id': 'yasaura', 'count': 6}
{'_id': 'Alecs01', 'count': 6}
{'_id': 'stucki1', 'count': 6}
{'_id': 'vt100', 'count': 6}
{'_id': 'GEOIG', 'count': 6}
{'_id': 'presakha', 'count': 6}
{'_id': 'oldtopos', 'count': 6}
{'_id': 'Carlos Ospina', 'count': 6}
{'_id': 'Whitehare', 'count': 6}
{'_id': 'schanillo', 'count': 6}
{'_id': 'snood1205', 'count': 6}
{'_id': 'DWandelt', 'count': 6}
{'_id': 'Andre Engels', 'count': 6}
{'_id': 'assalea', 'count': 6}
{'_id': 'gspr', 'count': 6}
{'_id': 'gcjunge', 'count': 6}
{'_id': 'nmmarri', 'count': 6}
{'_id': 'TravGW', 'count': 5}
{'_id': 'HPX1', 'count': 5}
{'_id': 'rolandg', 'count': 5}
{'_id': 'VLD158', 'count': 5}
{'_id': 'Nehaj', 'count': 5}
{'_id': 'nihalank', 'count': 5}
{'_id': 'GoWestTravel', 'count': 5}
{'_id': 'Keshann', 'count': 5}
```

```
{'_id': 'Fivea', 'count': 5}
{'_id': 'Walter Schlögl', 'count': 5}
{'_id': 'ryanparsons', 'count': 5}
{'_id': 'OSMF Redaction Account', 'count': 5}
{'_id': 'VLD178', 'count': 5}
{'_id': 'Evan Stumpf', 'count': 5}
{'_id': 'ToeBee', 'count': 5}
{'_id': 'vplauski_lyft', 'count': 5}
{'_id': 'Steve Ch', 'count': 5}
{'_id': 'krisrea29', 'count': 5}
{'_id': 'Segelpaule', 'count': 5}
{'_id': 'cpligovka_lyft', 'count': 5}
{'_id': 'svance92', 'count': 5}
{'_id': 'DowntownAbby', 'count': 5}
{'_id': 'grossing', 'count': 5}
{'_id': 'GenHubert', 'count': 5}
{'_id': 'VLD154', 'count': 5}
{'_id': 'bogdandr1', 'count': 5}
{'_id': 'VLD083', 'count': 5}
{'_id': 'Joseph Bollhofer', 'count': 5}
{'_id': 'beckyg1029', 'count': 5}
{'_id': 'brucewent', 'count': 5}
{'_id': 'daderkina', 'count': 5}
{'_id': 'mihaii_telenav', 'count': 5}
{'_id': 'Wyatt Maccarella', 'count': 5}
{'_id': 'yehster', 'count': 5}
{'_id': 'Hjart', 'count': 5}
{'_id': 'rprescott', 'count': 5}
{'_id': 'Megamega53', 'count': 5}
{'_id': 'KRizzitello', 'count': 5}
{'_id': 'vaibkaus', 'count': 5}
{'_id': 'OMapper', 'count': 5}
{'_id': 'cityeditor1000', 'count': 5}
{'_id': 'BugBuster', 'count': 5}
{'_id': 'gvarma', 'count': 5}
{'_id': 'yasponam', 'count': 5}
{'_id': 'alexsoldatkin96', 'count': 5}
{'_id': 'ajacobson', 'count': 5}
{'_id': 'pavapati', 'count': 5}
{'_id': 'baggg', 'count': 4}
{'_id': 'dkratzert', 'count': 4}
{'_id': 'ddarovsky', 'count': 4}
{'_id': 'JulienBalas', 'count': 4}
{'_id': 'PlaneMad', 'count': 4}
{'_id': 'SWF8', 'count': 4}
{'_id': 'ranjithjoy', 'count': 4}
{'_id': 'yayforfood', 'count': 4}
{'_id': 'Aleks-Berlin', 'count': 4}
{'_id': 'RiseUp26', 'count': 4}
{'_id': 'dylancassidy3', 'count': 4}
{'_id': 'akuksouski_lyft', 'count': 4}
{'_id': 'grink', 'count': 4}
{'_id': 'MLandis22', 'count': 4}
{'_id': 'thechubbygod', 'count': 4}
{'_id': 'TorCguy', 'count': 4}
{'_id': 'notehagur', 'count': 4}
{'_id': 'Olyon', 'count': 4}
```

```
{'_id': 'kickgraphics', 'count': 4}
{'_id': 'Snakebite', 'count': 4}
{'_id': 'Premsakhare', 'count': 4}
{'_id': 'SP!KE', 'count': 4}
{'_id': 'NK7272', 'count': 4}
{'_id': 'zors1843', 'count': 4}
{'_id': 'marianp_telenav', 'count': 4}
{'_id': 'nthdesign', 'count': 4}
{'_id': 'Ed Kwiatkowski', 'count': 4}
{'_id': 'Ericles', 'count': 4}
{'_id': 'sonmanek', 'count': 4}
{'_id': 'mneko', 'count': 4}
{'_id': 'hobbesvsboyle', 'count': 4}
{'_id': 'Dana Bongiovanni', 'count': 4}
{'_id': 'GreenRunner0', 'count': 4}
{'_id': 'VLD177', 'count': 4}
{'_id': 'lucsalmons', 'count': 4}
{'_id': 'bobbeyer', 'count': 4}
{'_id': 'Glenstorm_98', 'count': 4}
{'_id': 'Longford Landscape', 'count': 4}
{'_id': '3xodus', 'count': 4}
{'_id': 'JaLooooNz', 'count': 4}
{'_id': 'kli_lyft', 'count': 4}
{'_id': 'Manu Balusu', 'count': 4}
{'_id': 'noonito', 'count': 4}
{'_id': 'donsmith', 'count': 4}
{'_id': 'cafelatte', 'count': 4}
{'_id': 'hutchy68', 'count': 4}
{'_id': 'ShinkarevPavel', 'count': 4}
{'_id': 'Darren Wiebe - Aleph', 'count': 4}
{'_id': 'KR-KRKR-KR', 'count': 4}
{'_id': 'ChaseChris', 'count': 4}
{'_id': 'Bryce C Nesbitt', 'count': 3}
{'_id': 'van Rees', 'count': 3}
{'_id': 'wolfgang8741', 'count': 3}
{'_id': 'ramyaragupathy', 'count': 3}
{'_id': 'tom127106', 'count': 3}
{'_id': 'mdroads', 'count': 3}
{'_id': 'wasat', 'count': 3}
{'_id': 'djalterego', 'count': 3}
{'_id': 'Dami_Tn', 'count': 3}
{'_id': 'DougMoffitt', 'count': 3}
{'_id': 'VLD152', 'count': 3}
{'_id': 'VLD036', 'count': 3}
{'_id': 'UE_Su', 'count': 3}
{'_id': 'VlIvYur', 'count': 3}
{'_id': 'woodpeck', 'count': 3}
{'_id': 'folgers', 'count': 3}
{'_id': 'VLD079', 'count': 3}
{'_id': 'twra', 'count': 3}
{'_id': 'REWNICK', 'count': 3}
{'_id': 'Forest Gregg', 'count': 3}
{'_id': 'RobH', 'count': 3}
{'_id': 'nathan206', 'count': 3}
{'_id': 'navymoves', 'count': 3}
{'_id': 'Nandyala', 'count': 3}
{'_id': 'abviola', 'count': 3}
```

```
{'_id': 'rawatg', 'count': 3}
{'_id': 'VLD032', 'count': 3}
{'_id': 'Some_Person', 'count': 3}
{'_id': 'wambacher', 'count': 3}
{'_id': 'KevinWalshe', 'count': 3}
{'_id': 'ewliebig', 'count': 3}
{'_id': 'germyb', 'count': 3}
{'_id': 'Flir', 'count': 3}
{'_id': 'woodpeck_repair', 'count': 3}
{'_id': 'Mateusz Konieczny - bot account', 'count': 3}
{'_id': 'salix01', 'count': 3}
{'_id': 'Rps333', 'count': 3}
{'_id': 'curtarledge', 'count': 3}
{'_id': 'Spartan Pools And Spas', 'count': 3}
{'_id': 'Alex Wauck', 'count': 3}
{'_id': 'rdanastorg', 'count': 3}
{'_id': 'EZRouting', 'count': 3}
{'_id': 'John Peterson', 'count': 2}
{'_id': 'PGW77', 'count': 2}
{'_id': 'maddiegogo', 'count': 2}
{'_id': 'vssms', 'count': 2}
{'_id': 'mb32', 'count': 2}
{'_id': 'Claudius Henrichs', 'count': 2}
{'_id': 'AlexCampean', 'count': 2}
{'_id': 'Pizza15', 'count': 2}
{'_id': 'VLD168', 'count': 2}
{'_id': 'claralyn', 'count': 2}
{'_id': 'davidearl', 'count': 2}
{'_id': 'bbmiller', 'count': 2}
{'_id': 'SenseZeroFloat', 'count': 2}
{'_id': 't3nnispro123', 'count': 2}
{'_id': 'mcnabbian', 'count': 2}
{'_id': 'ruthmaben', 'count': 2}
{'_id': 'Skal', 'count': 2}
{'_id': 'dannmer', 'count': 2}
{'_id': 'jake wise', 'count': 2}
{'_id': 'gdt', 'count': 2}
{'_id': 'kjon', 'count': 2}
{'_id': 'MapGirl777', 'count': 2}
{'_id': 'johnament', 'count': 2}
{'_id': 'nyuriks', 'count': 2}
{'_id': 'owenwastaken', 'count': 2}
{'_id': 'Peter14', 'count': 2}
{'_id': 'merz', 'count': 2}
{'_id': 'LostAmericana', 'count': 2}
{'_id': 'mstriewe', 'count': 2}
{'_id': 'scai', 'count': 2}
{'_id': 'coleman', 'count': 2}
{'_id': 'VLD155', 'count': 2}
{'_id': 'kisaa', 'count': 2}
{'_id': 'Hb-', 'count': 2}
{'_id': 'pabi12', 'count': 2}
{'_id': 'Larry Butler', 'count': 2}
{'_id': 'Manu1400', 'count': 2}
{'_id': 'myMapGil', 'count': 2}
{'_id': 'KBruce', 'count': 2}
{'_id': 'Naren_PR', 'count': 2}
```

```
{'_id': 'ben4244', 'count': 2}
{'_id': 'Apuq', 'count': 2}
{'_id': 'Rvansaun', 'count': 2}
{'_id': 'HubMiner', 'count': 2}
{'_id': 'Bootprint', 'count': 2}
{'_id': 'Skybunny', 'count': 2}
{'_id': 'mapman44', 'count': 2}
{'_id': 'ninjamask', 'count': 2}
{'_id': 'TinekeThio', 'count': 2}
{'_id': 'Matthew Lawson', 'count': 2}
{'_id': 'VLD171', 'count': 2}
{'_id': 'Kent Shaffer', 'count': 2}
{'_id': 'acocaracha', 'count': 2}
{'_id': 'VLD175', 'count': 2}
{'_id': 'AndreThib', 'count': 2}
{'_id': 'wikimax', 'count': 2}
{'_id': 'City Print', 'count': 2}
{'_id': 'fixedbusiness', 'count': 2}
{'_id': 'oliviap_telenav', 'count': 2}
{'_id': 'sriharsd', 'count': 2}
{'_id': 'Billh999', 'count': 2}
{'_id': 'Kla_Ger', 'count': 2}
{'_id': 'anddmsg', 'count': 1}
{'_id': 'Springskies', 'count': 1}
{'_id': 'Nakaner-repair', 'count': 1}
{'_id': "Sujit D'Mello", 'count': 1}
{'_id': 'springer2003', 'count': 1}
{'_id': 'Itd5sS6uZ', 'count': 1}
{'_id': 'kinghowdy', 'count': 1}
{'_id': 'maggot27', 'count': 1}
{'_id': 'justinnj', 'count': 1}
{'_id': 'charabor', 'count': 1}
{'_id': 'Danks2006', 'count': 1}
{'_id': 'welcometonj', 'count': 1}
{'_id': 'dfredrick', 'count': 1}
{'_id': 'weronika5', 'count': 1}
{'_id': 'gilbane', 'count': 1}
{'_id': 'AnonymousAlligator', 'count': 1}
{'_id': 'rob216', 'count': 1}
{'_id': 'rs1016', 'count': 1}
{'_id': 'dcpaschall', 'count': 1}
{'_id': 'dforsi', 'count': 1}
{'_id': 'gpserror', 'count': 1}
{'_id': 'Rajamani David', 'count': 1}
{'_id': 'Horizondisp', 'count': 1}
{'_id': 'Andrew Fish', 'count': 1}
{'_id': 'sebastic', 'count': 1}
{'_id': 'Land-o Calrissian', 'count': 1}
{'_id': 'ai1', 'count': 1}
{'_id': 'Shashank Saraogi', 'count': 1}
{'_id': 'NielsEricNielsen', 'count': 1}
{'_id': 'InsertUser', 'count': 1}
{'_id': 'daniel_solow', 'count': 1}
{'_id': 'karpfish', 'count': 1}
{'_id': 'Liz Porcelli', 'count': 1}
{'_id': 'HolgerJeromin', 'count': 1}
{'_id': 'McHughBDSTech', 'count': 1}
```

```
{'_id': 'SelfishSeahorse', 'count': 1}
{'_id': 'anukrits', 'count': 1}
{'_id': 'Syed Mujeeb', 'count': 1}
{'_id': 'Jim Mehling', 'count': 1}
{'_id': 'ebt12', 'count': 1}
{'_id': 'isca31', 'count': 1}
{'_id': 'ReadyTravellerOne', 'count': 1}
{'_id': 'xybot', 'count': 1}
{'_id': 'R0bst3r', 'count': 1}
{'_id': 'Rahul Dantu', 'count': 1}
{'_id': 'Margori123', 'count': 1}
{'_id': 'tjon', 'count': 1}
{'_id': 'mosleykl', 'count': 1}
{'_id': 'xliu_lyft', 'count': 1}
{'_id': 'arehrlich', 'count': 1}
{'_id': 'donaldschafer', 'count': 1}
{'_id': 'Carl Simonson', 'count': 1}
{'_id': 'Mouameme', 'count': 1}
{'_id': 'RedBarron067', 'count': 1}
{'_id': 'keinseier', 'count': 1}
{'_id': 'Kerbaswh', 'count': 1}
{'_id': 'impiaaa', 'count': 1}
{'_id': 'djohnso84', 'count': 1}
{'_id': 'Mrhoopy', 'count': 1}
{'_id': 'achims311', 'count': 1}
{'_id': 'Nick_Springer', 'count': 1}
{'_id': 'rowers2', 'count': 1}
{'_id': 'vng_me', 'count': 1}
{'_id': '10cannonbolt', 'count': 1}
{'_id': 'xogalla', 'count': 1}
{'_id': 'brucelthrockmor', 'count': 1}
{'_id': "A'Omare", 'count': 1}
{'_id': 'HHCacher', 'count': 1}
{'_id': 'Stephen214', 'count': 1}
{'_id': 'Guylamar2006', 'count': 1}
{'_id': 'h4ck3rm1k3', 'count': 1}
{'_id': 'joshcartagena', 'count': 1}
{'_id': 'Kayla Colflesh', 'count': 1}
{'_id': 'Heinz_V', 'count': 1}
{'_id': 'Daway', 'count': 1}
{'_id': 'Evgeni Sautin', 'count': 1}
{'_id': 'chrisk84', 'count': 1}
{'_id': 'beej71', 'count': 1}
{'_id': 'Christopher-0118', 'count': 1}
{'_id': 'Pipe303', 'count': 1}
{'_id': 'bhietsch', 'count': 1}
{'_id': 'Name Not Taken', 'count': 1}
{'_id': 'xunilOS', 'count': 1}
{'_id': 'dwirkijowski', 'count': 1}
{'_id': 'Fatte', 'count': 1}
{'_id': 'SL38', 'count': 1}
{'_id': 'alivelis', 'count': 1}
{'_id': 'Friendly_Ghost', 'count': 1}
{'_id': 'Chris Jones131', 'count': 1}
{'_id': 'Lucitucci', 'count': 1}
{'_id': 'Anujain100', 'count': 1}
{'_id': 'MapletonResident', 'count': 1}
```

```
{'_id': 'keverets', 'count': 1}
{'_id': 'wambag', 'count': 1}
{'_id': 'CloCkWeRX', 'count': 1}
{'_id': 'BigL900', 'count': 1}
{'_id': 'mburke8', 'count': 1}
{'_id': 'hendrik-17', 'count': 1}
{'_id': 'KD2PM', 'count': 1}
{'_id': 'robhedrick', 'count': 1}
{'_id': 'Jeff Freedman', 'count': 1}
{'_id': 'Micduncan', 'count': 1}
{'_id': 'MECoonMama', 'count': 1}
{'_id': 'westendguy', 'count': 1}
{'_id': 'FIM', 'count': 1}
{'_id': 'NickBolten', 'count': 1}
{'_id': 'michelleblassou', 'count': 1}
{'_id': 'parth624', 'count': 1}
{'_id': 'TheBlackMan', 'count': 1}
{'_id': 'pyrog', 'count': 1}
{'_id': 'Barry Higgins', 'count': 1}
{'_id': 'jose_sousa', 'count': 1}
{'_id': 'aftonlimo', 'count': 1}
{'_id': 'MattForce', 'count': 1}
{'_id': 'kitchoi', 'count': 1}
{'_id': 'kasbadri', 'count': 1}
{'_id': 'AxelBoldt', 'count': 1}
{'_id': 'OSM_Rulez', 'count': 1}
{'_id': 'oini', 'count': 1}
{'_id': 'cdavila', 'count': 1}
{'_id': 'jwdefer', 'count': 1}
{'_id': 'shalomtherape', 'count': 1}
{'_id': 'jenkins1967', 'count': 1}
{'_id': '6886', 'count': 1}
{'_id': 'shawat94', 'count': 1}
{'_id': 'HiddeWie', 'count': 1}
{'_id': 'pilot8766', 'count': 1}
{'_id': 'Traderiskgroup', 'count': 1}
{'_id': 'derFred', 'count': 1}
{'_id': 'BritBloke', 'count': 1}
{'_id': 'Gwp19', 'count': 1}
{'_id': 'GISJoe', 'count': 1}
{'_id': 'user1923', 'count': 1}
{'_id': 'zeromap', 'count': 1}
{'_id': 'sam1234567', 'count': 1}
{'_id': 'seattlefyi', 'count': 1}
{'_id': 'VitKlim', 'count': 1}
{'_id': 'dankpoet', 'count': 1}
{'_id': 'yasse1406', 'count': 1}
{'_id': 'James R Baxter', 'count': 1}
{'_id': 'Akkuhappo', 'count': 1}
{'_id': 'Letter Of Power', 'count': 1}
{'_id': 'storeyme', 'count': 1}
{'_id': 'adamos', 'count': 1}
{'_id': 'gormur', 'count': 1}
{'_id': 'Dero Bike Racks', 'count': 1}
{'_id': 'davidfaure_bot', 'count': 1}
{'_id': 'Rincewind7', 'count': 1}
{'_id': 'Pavel Chyzhonak', 'count': 1}
```

```
{'_id': 'almiki', 'count': 1}
{'_id': 'peterguy', 'count': 1}
{'_id': 'SanjevRajaram', 'count': 1}
{'_id': 'weg', 'count': 1}
{'_id': 'cpoan', 'count': 1}
{'_id': 'Steve', 'count': 1}
{'_id': 'Donald Schafer', 'count': 1}
{'_id': 'Bruce Barlow', 'count': 1}
{'_id': 'Josef73', 'count': 1}
{'_id': 'GRailMapper', 'count': 1}
{'_id': 'philherbert', 'count': 1}
{'_id': 'raphaelmirc', 'count': 1}
{'_id': 'theneilc', 'count': 1}
{'_id': 'smerzzz', 'count': 1}
```

## Section 3: Additional Ideas

Contributor statistics and gamification suggestion: The contributions of users seems incredibly skewed. Here are some user percentage statistics:

Top user contribution percentage (“iskra32”) - 26% Combined top 2 users' contribution (“iskra32”)” and “OceanVortex”) - 42% Combined Top 10 users contribution - 62.20% Users making up only 5% of posts (1/51079)

Looking into these user percentages, I believe “gamification” is a motivating factor for contribution. In the context of the OpenStreetMap, if users data are more accurately displayed then more people will take interest in submitting edits to the map data.

## Section 4: Conclusion

After this review of the data it’s obvious that the Trenton city area is incomplete, though I have cleaned up well for the purpose of this project. However, the improvement may bring more issues if not implemented correctly:

1.Gamification may impact the quality of the data submitted from the contributors. We should keep in mind quality is more important then quantity.

2.If the difference between the highest score and other scores is too large, users may easily loose their interest. Therefore, we should implement the logic in such a way that as the scores rise high, it should become more difficult to increase.

With a robust data processor, it would possibly input more cleaner data to OpenStreetMap.org.